

Teknik Cepat Pengembangan **MULTIMEDIA INTERAKTIF**

dengan **MPI Component**



Teknik Cepat Pengembangan Multimedia Interaktif

Dr. Wandah Wibawanto, S.Sn., M.Ds.

Diterbitkan oleh



IKAPI No.026/Anggota Luar Biasa/JTE/2021

APPTI No. 003.045.1.05.2018



Hak Cipta © pada penulis dan dilindungi Undang-Undang
Penerbitan. Hak Penerbitan pada UNNES PRESS.

Dicetak oleh **UNNES Press**.

Jl. Kelud Raya No. 2 Semarang 50237

Telp. (024) 86008700 ext. 062

Dilarang mengutip sebagian atau seluruh isi buku ini dalam bentuk apapun tanpa izin
dari penerbit.

TEKNIK CEPAT PENGEMBANGAN MULTIMEDIA INTERAKTIF

Penulis:

Dr. Wandah Wibawanto, S.Sn. M.Ds.

Desain, Pemrograman & Tata Letak:

Dr. Wandah Wibawanto, S.Sn. M.Ds.

21,5 x 28 cm (xvii + 203 Halaman)

Cetakan Pertama, 2023

ISBN 978-602-285-378-7

Sanksi Pelanggaran Pasal 72 Undang-undang Nomor 19 Tahun 2002 Tentang Hak Cipta

1. Barangsiapa dengan sengaja melanggar dan tanpa hak melakukan perbuatan sebagaimana dimaksud dalam pasal 2 ayat (1) atau pasal 49 ayat (1) dan ayat (2) dipidana dengan pidana penjara masing-masing paling singkat 1 (satu) bulan dan/atau denda paling sedikit Rp. 1.000.000,00 (satu juta rupiah), atau pidana penjara paling lama 7 (tujuh) tahun dan/ atau denda paling banyak Rp. 5.000.000.000,00 (lima Milyar).
2. Barangsiapa dengan sengaja menyiarkan, memamerkan, mengedarkan atau menjual, kepada umum suatu ciptaan atau barang hasil pelanggaran Hak Cipta sebagaimana dimaksud dalam ayat (1) dipidana dengan pidana penjara paling lama 5 (lima) tahun dan/atau denda paling banyak Rp. 50.000.000,00 (limapuluh juta rupiah).

KATA PENGANTAR

Alhamdulillah robbil'aalamin, pada akhirnya buku ini selesai ditulis untuk mengawali tahun 2023. Rasa syukur yang terdalam saya sampaikan karena diberikan kemudahan dalam penulisan buku ini. Buku ini saya tulis untuk melengkapi buku sebelumnya dan untuk memberikan gambaran yang komprehensif terkait pengembangan multimedia interaktif.

Multimedia interaktif menjadi salah satu mata kuliah yang selalu saya sampaikan setiap tahunnya. Sejak memasuki dunia akademik yaitu pada tahun 2014, sepanjang mengampu mata kuliah multimedia interaktif begitu banyak pengalaman terkait proses belajar mengembangkan multimedia interaktif. Pengalaman dua dekade di dunia industri, khususnya di dunia pemrograman aplikasi pada kenyataannya sangat sulit untuk diimplementasikan secara langsung kepada peserta didik. Dunia akademik seolah jauh tertinggal beberapa langkah dibandingkan dengan dunia industri jika berbicara terkait pengembangan aplikasi, khususnya aplikasi multimedia. Untuk mengejar ketertinggalan tersebut tentunya membutuhkan suatu usaha khusus agar peserta didik yang sebagian besar adalah pemula mampu untuk mengikutinya.

Buku pengantar terkait Desain dan Pemrograman Multimedia Pembelajaran interaktif yang saya tulis pada tahun 2017 menjadi salah satu acuan pengembangan aplikasi multimedia interaktif, namun masih memiliki begitu banyak celah ketika dihadapkan dengan keilmuan desain dan pemrograman. Oleh karena itu, saya ingin sekali menutupi celah-celah tersebut dengan menuliskan sebuah buku baru yang lebih komprehensif dan lebih mudah untuk diimplementasikan oleh pembelajar pemula. Pengembangan menggunakan *compiled clip* yang telah diimplementasikan dalam kelas multimedia selama satu tahun terakhir, terbukti mempermudah para pembelajar pemula khususnya mahasiswa desain untuk mengembangkan aplikasi-aplikasi multimedia interaktif yang menarik, interaktif, dan fungsional. Oleh karena itu buku ini ditulis sebagai panduan bagi pembaca untuk mengembangkan multimedia interaktif dengan bantuan *compiled clip* atau dalam buku ini disebut sebagai *MPI-Component*.

Buku ini saya tulis menjadi beberapa bab, menggunakan pendekatan pengembangan aplikasi yang saya simpulkan dari beberapa literatur. Setiap bab nya membahas proses pengembangan aplikasi multimedia interaktif secara bertahap, dimulai dari perencanaan, proses desain berikut teori-teori desain yang terkait, proses pemograman, proses ujicoba sampai dengan proses publishing. Dalam buku ini, saya berusaha menjelaskan sedetail mungkin proses yang dimaksud, namun saya menyadari bahwa informasi deskriptif yang disampaikan melalui buku dalam beberapa kasus sulit untuk diikuti. Oleh karena itu, tutorial-tutorial yang ada di dalam buku ini saya integrasikan dengan tutorial yang ada di website personal saya dan channel youtube saya. Hal ini ditujukan untuk saling melengkapi antara literatur tertulis dengan media audio visual.

Terimakasih saya sampaikan kepada seluruh mahasiswa mata kuliah multimedia interaktif DKV Universitas Negeri Semarang, yang dalam buku ini saya jadikan sebagai sampel produk. Saya memohon maaf yang sebesar-besarnya apabila terdapat kesalahan atau kekeliruan dalam buku ini, dan berharap agar buku ini membawa manfaat bagi pembaca dan berkontribusi dalam perkembangan multimedia interaktif di Indonesia.

Januari 2023,

Penulis

DAFTAR ISI

Kata Pengantar	iii
Daftar Isi.....	v
Daftar Gambar	ix
Daftar Tabel	xvii
 BAB I Konsep Dasar Multimedia Interaktif	
1.1 Definisi.....	1
1.2 Komponen Multimedia Interaktif.....	2
1.3 Metode Pengembangan Multimedia Interaktif	2
1.4 Tujuan dan Target Buku.....	6
1.5 Aplikasi dan Contoh Multimedia Interaktif	6
1.6 Distribusi Buku	7
 BAB 2 <i>Design Treatment</i>	
2.1 Multimedia sebagai bagian dari desain.....	9
2.2 Unsur Desain.....	10
2.3 Prinsip Desain	12
2.3 Merencanakan Multimedia Interaktif.....	13
2.4 Pengelolaan sumber daya dan perencanaan jadwal kerja	16
 BAB 3 <i>Visual Development Phase</i>	
Desain Antar Muka (<i>User Interface</i>)	
3.1 Desain Antarmuka (<i>User Interface</i>)	19
3.2 Elemen Antar Muka	28
3.3 Tata Letak (<i>Layout</i>)	38
3.3.1 <i>Aspect Ratio</i>	41
3.3.2 Sistem <i>Grid</i>	42
3.3.3 <i>Eyeflow</i>	46
3.3.4 <i>Fine Layout</i> halaman multimedia interaktif.....	49
 BAB 4 Adobe Animate	
4.1 Area Kerja Adobe Animate.....	71
4.1.1 Menu Utama	73
4.1.2 <i>Tools</i>	73
4.1.3 <i>Timeline</i>	74
4.1.4 <i>Stage</i>	76
4.1.5 <i>Properties</i>	76
4.1.6 <i>Library</i>	77

4.1.7 Action Panel	77
4.2 Membuat Proyek Baru dengan Adobe Animate	78
4.3 Objek dan <i>Symbol</i>	79
4.3.1 <i>MovieClip</i>	82
4.3.2 Tombol	85
4.3.3 <i>Graphic</i>	88
4.3.4 <i>Bitmap</i>	89
4.3.5 Suara	90
4.3.6 Video	92
4.4 Animasi	92
4.5 <i>Instance name</i>	92
4.6 <i>Linkage</i>	93

BAB 5 Visual Development Phase

Menyusun Halaman Multimedia Interaktif

5.1 Format Proyek (Susunan Layer)	97
5.2 Penyusunan Halaman	99
5.3 Penambahan <i>Instance name</i>	106

BAB 6 Programming Phase

Pengembangan Multimedia Interaktif dengan *MPI Component*

6.1 Konsep dasar <i>MPI Component</i>	107
6.2 Menambahkan <i>MPI Component</i> ke <i>file</i> proyek multimedia interaktif	108
6.3 Navigasi Halaman	110
6.4 Penambahan Halaman	112
6.5 Transisi Halaman	115
6.5.1 Transisi dengan Efek <i>MovieClip</i>	115
6.6 <i>Scroll</i> konten	120
6.6.1 Penjelasan Kode	124
6.7 Galeri	124
6.7.1 Penjelasan Program	129
6.7.2 Menambahkan Efek Pergeseran pada Galeri	129

BAB 7 Programming Phase

Suara dan Video

7.1 Penambahan Suara	131
7.2 Menambahkan Suara Tombol	133
7.3 Menambahkan Fitur <i>On-Off</i> Suara	134
7.4 Menambahkan Fitur Pengaturan Volume Suara	135
7.5 Penambahan Video	137

BAB 8 Fitur Pelengkap Multimedia Interaktif

8.1 Intro	141
8.2 <i>Popup</i>	144
8.3 <i>Login</i>	148
8.4 <i>Drag and drop</i>	149
8.4.1 Menambahkan Halaman Permainan	149
8.4.2 Permainan <i>Drag and drop</i>	151
8.5 Kuis	156
8.5.1 Membuat <i>database</i> soal untuk kuis	158
8.5.2 Perrograman kuis	159
8.5.3 Penjelasan Program	160
8.6 <i>Puzzle</i>	161
8.7 <i>Timer</i>	164
8.8 <i>Kursor Mouse</i>	166

BAB 9 Virtual Lab

9.1 Laboratorium Virtual	167
9.1.1 Menyiapkan Halaman Simulasi	167
9.2 Simulasi Panas	169
9.3 Interaktivitas pada Simulasi	172
9.3.1 Mengatur Volume Zat Cair dengan <i>Slider</i>	172
9.3.2 Menambahkan Gravitasi pada Objek	173
9.3.3 Menambahkan Pengaturan Isi Gelas	173
9.4 Penambahan Rumus pada Simulasi	175
9.4.1 Penjelasan Program	176
9.5 Pengembangan Simulasi Lain	177

BAB 10 Publishing Phase

10.1 Konsep Dasar Multiplatform	179
10.2 Desktop (.exe)	180
10.3 Android	181
10.3.1 AIR SDK	182
10.3.2 Publikasi dengan Output APK	183
10.3.3 Mengujicoba File APK pada Gawai	187
10.3.4 Mode Layar Penuh Android	188
10.3.5 Publikasi Bertipe AAB (Android App Bundle) untuk Play Store	190
10. 3. 5.1 Instalasi Android Studio	191
10. 3. 5.2 Mengatur Konfigurasi AIR SDK	192
10. 3. 5.3 Menghasilkan File bertipe AAB	194
10.4 Platform Web Browser	194

BAB 11 Penutup	199
Daftar Pustaka.....	201
Tentang Penulis.....	203

DAFTAR GAMBAR

Gambar 1. peta profesi DKV	9
Gambar 2. dokumentasi multimedia pembelajaran interaktif.....	14
Gambar 3. rencana halaman awal multimedia interaktif “Batik Kita”	14
Gambar 4. rencana sub halaman multimedia interaktif “Batik Kita”	15
Gambar 5. rencana halaman awal multimedia interaktif “fire fight”	16
Gambar 6. penjadwalan kerja tim pengembang aplikasi.....	17
Gambar 7. tampilan halaman “Vegefruits”	20
Gambar 8. tampilan halaman “Flower Garden”	21
Gambar 9. multimedia “Satwa Endemik Indonesia”	23
Gambar 10. multimedia interaktif “Sinau Batik”	24
Gambar 11. aplikasi Marbel Panduan Puasa Ramadhan	25
Gambar 12. multimedia “Inside Our Body”	26
Gambar 13. aplikasi multimedia interaktif “Cerita Afan”	27
Gambar 14. <i>input control</i> multimedia interaktif “Inside Our Body”	28
Gambar 15. aplikasi multimedia interaktif “Cerita Afan”	29
Gambar 16. <i>virtual joystick</i> pada aplikasi Astro Farmer	29
Gambar 17. tombol tindakan pada multimedia interaktif “Bioma”	30
Gambar 18. multimedia interaktif “Cara Menjadi Youtubers”	31
Gambar 19. tombol teks pada multimedia interaktif “Tur Planet”	32
Gambar 20. tombol transparan “Start with email”	33
Gambar 21. tombol <i>onOff</i> suara.....	33
Gambar 22. peletakan tombol <i>toggle on off</i> suara	33
Gambar 23. komponen informasi pada multimedia “Rempah Indonesia”	35
Gambar 24. panel kartu informasi pada multimedia interaktif “Vegefruit”	36
Gambar 25. panel <i>scroll</i> pada multimedia “Mengenal Lapisan Bumi”	36
Gambar 26. formulir pada “onsertify”.....	37
Gambar 27. permainan <i>drag and drop</i> pada multimedia “Cerita Afan”	37
Gambar 28. satu set permainan dalam multimedia “Bioma”	38
Gambar 29. prinsip <i>proximity</i>	39
Gambar 30. prinsip <i>similarity</i>	39
Gambar 31. prinsip <i>continuity</i>	39
Gambar 32. prinsip <i>closure</i>	40
Gambar 33. Rubin Vas yang menunjukkan hubungan <i>figure-ground</i>	40
Gambar 34. keterhubungan	40
Gambar 35. multimedia interaktif “Panduan Umrah” dengan posisi <i>portrait</i>	41
Gambar 36. multimedia interaktif “Bendera Dunia” dengan posisi <i>landscape</i>	42
Gambar 37. aspek rasio	42
Gambar 38. <i>rule of third</i> pada multimedia interaktif “Batik Kita”	43

Gambar 39. finalisasi penerapan <i>rule of third</i>	43
Gambar 40. penggunaan <i>grid</i> kolom tunggal	44
Gambar 41. penggunaan <i>grid</i> multikolom	45
Gambar 42. penggunaan <i>grid</i> modular	46
Gambar 43. <i>eyeflow</i> dengan pendekatan diagram Gutenberg.....	47
Gambar 44. <i>eyeflow</i> dengan pola-F	47
Gambar 45. <i>eyeflow</i> dengan pola-Z	48
Gambar 46. multimedia interaktif “Menenal P3K” dan “4 Sehat 5 Sempurna” ...	48
Gambar 47. Multimedia “Batik Kita”	49
Gambar 48. multimedia “Bersih-bersih”	50
Gambar 49. multimedia “Ayo Memilah Sampah”	51
Gambar 50. multimedia “Fire Fight”	52
Gambar 51. multimedia interaktif “Kuliner Khas Nusantara”	53
Gambar 52. multimedia interaktif “Menenal Tokoh Uang”	54
Gambar 53. multimedia interaktif “Menenal P3K”	55
Gambar 54. multimedia interaktif “Know Your Camera”	56
Gambar 55. multimedia interaktif “Vegefruit”	57
Gambar 56. multimedia interaktif “Bersih-bersih Yuk”	58
Gambar 57. multimedia interaktif “Menjelajah Bumi Raflesia”	59
Gambar 58. multimedia interaktif “Rempah Indonesia”	60
Gambar 59. multimedia interaktif “Sinai Batik”	61
Gambar 60. multimedia interaktif “Ayo Belajar Tata Surya”	62
Gambar 61. multimedia interaktif “Piliana Cultourism”	63
Gambar 62. multimedia interaktif “Ngigel”	64
Gambar 63. multimedia interaktif “Lincah Matika”	65
Gambar 64. multimedia interaktif “I am a mammal”	66
Gambar 65. multimedia interaktif “Menenal Bioma”	67
Gambar 66. multimedia interaktif “Scuba Diving”	68
Gambar 67. multimedia interaktif “Pahlawan Nasional Indonesia”	69
Gambar 68. multimedia interaktif “Pakaian dan Rumah Adat Nusantara”	70
Gambar 69. area kerja Adobe Animate pada mode <i>esensial</i>	71
Gambar 70. lokasi pengaturan area kerja (<i>workspaces</i>)	72
Gambar 71. opsi mode <i>classic</i>	72
Gambar 72. tampilan pada mode <i>classic</i>	72
Gambar 73. Menu Utama	73
Gambar 74. <i>Tools</i>	73
Gambar 75. elemen grafis vektor	73
Gambar 76. <i>Timeline</i>	74
Gambar 77. <i>Keyframe</i>	74
Gambar 78. <i>Blank Keyframe</i>	74
Gambar 79. <i>Frame</i> berdurasi 1 detik.....	75

Gambar 80. <i>Keyframe</i> dengan kode <i>Actionscript</i>	75
Gambar 81. <i>motion tween</i>	75
Gambar 82. <i>Frame</i> dengan suara.....	75
Gambar 83. <i>Stage</i>	76
Gambar 84. panel <i>Properties</i>	76
Gambar 85. panel <i>Library</i>	77
Gambar 86. panel <i>Action</i>	77
Gambar 87. membuat <i>File</i> baru.....	78
Gambar 88. panel <i>New Document</i>	79
Gambar 89. membuat kotak.....	79
Gambar 90. menyeleksi gambar.....	80
Gambar 91. menyeleksi <i>stroke</i>	80
Gambar 92. menyeleksi <i>fill</i>	80
Gambar 93. mengedit titik dengan <i>subselection tool</i>	81
Gambar 94. <i>Convert to Symbol</i>	81
Gambar 95. <i>Symbol</i> Kotak pada panel <i>Library</i>	81
Gambar 96. mode edit <i>symbol</i>	82
Gambar 97. membuat <i>MovieClip</i> burung.....	83
Gambar 98. mode edit <i>MovieClip</i> burung.....	83
Gambar 99. menambahkan <i>Blank Keyframe</i>	84
Gambar 100. membuat gambar pada <i>Frame 2</i>	84
Gambar 101. luaran <i>MovieClip</i> burung.....	84
Gambar 102. menggeser <i>Frame</i>	85
Gambar 103. menambahkan durasi dengan <i>insert Frame</i>	85
Gambar 104. membuat kotak.....	86
Gambar 105. menambahkan <i>Static Text</i>	86
Gambar 106. membuat <i>symbol</i> bertipe <i>button</i>	86
Gambar 107. <i>cursor</i> mouse di atas tombol.....	87
Gambar 108. struktur <i>Frame symbol</i> bertipe <i>button</i>	87
Gambar 109. menambahkan <i>Frame Over</i> pada tombol.....	87
Gambar 110. hasil <i>copypaste MovieClip</i> burung.....	88
Gambar 111. mengubah tipe menjadi <i>Graphic</i>	88
Gambar 112. menambahkan durasi animasi.....	89
Gambar 113. pengaturan <i>looping</i> pada <i>Graphic</i>	89
Gambar 114. mengimport <i>file</i>	90
Gambar 115. memilih gambar yang diimpor.....	90
Gambar 116. mengimpor suara.....	90
Gambar 117. tampilan gelombang suara pada <i>Timeline</i>	91
Gambar 118. <i>Properties</i> suara.....	91
Gambar 119. menambahkan <i>instance name</i>	93
Gambar 120. menambahkan kode.....	93

Gambar 121. pesan error ketika terjadi kesalahan <i>instance name</i>	93
Gambar 122. menambahkan <i>Linkage</i>	94
Gambar 123. struktur navigasi halaman multimedia interaktif “Tebak Satwa” ...	97
Gambar 124. membuat 4 buah <i>Layer</i>	98
Gambar 125. mengubah nama <i>Layer</i>	98
Gambar 126. <i>folder</i> gambar dan aset visual	99
Gambar 127. menambahkan <i>Blank Keyframe</i> pada <i>Frame 10 Layer background</i>	100
Gambar 128. mengimpor latar halaman judul.....	100
Gambar 129. peletakan latar tombol pada <i>Stage</i>	101
Gambar 130. mengubah gambar menjadi tombol.....	101
Gambar 131. mengatur jenis teks	102
Gambar 132. mengatur efek <i>outline</i> teks.....	102
Gambar 133. tombol “Jenis Hewan”	102
Gambar 134. mengimpor gambar ke <i>Library</i>	103
Gambar 135. mengubah <i>bitmap</i> latar tombol	103
Gambar 136. mengatur tombol “Jenis Hewan”	103
Gambar 137. menduplikasi simbol.....	104
Gambar 138. mengedit tombol “btn_video”	104
Gambar 139. tombol navigasi pada halaman judul.	104
Gambar 140. menambahkan halaman “jenis hewan”	105
Gambar 141. menambahkan konten halaman jenis satwa	105
Gambar 142. penambahan <i>instance name</i>	106
Gambar 143. penambahan <i>instance name</i>	106
Gambar 144. simbol MPI Component dalam format <i>precompiled clip</i>	108
Gambar 145. halaman <i>download</i> MPI-Component.....	108
Gambar 146. membuka <i>Library</i> file MPI-component fla.....	109
Gambar 147. menambahkan MPI Component.....	109
Gambar 148. menambahkan <i>Frame</i>	109
Gambar 149. Penambahan <i>instance name</i> pada simbol mpi.....	110
Gambar 150. representasi sebelum dan setelah penambahan kode	110
Gambar 151. menambahkan tombol <i>home</i> dari <i>Library</i> MPI-Component.....	111
Gambar 152. <i>flowchart</i> pengembangan multimedia interaktif.....	112
Gambar 153. struktur <i>Timeline</i> untuk penambahan halaman baru	113
Gambar 154. konten <i>Frame 30</i>	113
Gambar 155. halaman “hewan omnifora” dan “hewan herbivora”	114
Gambar 156. pesan kesalahan yang muncul akibat penulisan <i>instance name</i> yang belum dilakukan.....	114
Gambar 157. mengunci <i>Layer</i>	115
Gambar 158. posisi tirai untuk efek transisi <i>MovieClip</i>	116
Gambar 159. mengubah gambar menjadi <i>MovieClip</i>	116
Gambar 160. mendistribusikan <i>MovieClip</i> ke masing-masing <i>Layer</i>	117

Gambar 161. posisi objek di Frame 15	117
Gambar 162. menambahkan animasi <i>motion tween</i>	118
Gambar 163. animasi tirai menutup dan membuka.....	118
Gambar 164. menambahkan <i>Linkage</i>	119
Gambar 165. membuat informasi untuk fitur <i>scroll</i>	120
Gambar 166. struktur <i>MovieClip</i> “informasi karnivora”	121
Gambar 167. proses <i>masking</i>	122
Gambar 168. menambahkan <i>scroller</i>	122
Gambar 169. posisi <i>MovieClip scrollHMC</i> saat vertikal.....	123
Gambar 170. hasil penambahan fitur <i>scroll</i>	123
Gambar 171. Contoh fitur galeri dalam multimedia interaktif	124
Gambar 172. fitur galeri pada multimedia interaktif “Belajar kimia”	125
Gambar 173. file gambar hewan karnivora.....	125
Gambar 174. struktur <i>Timeline</i> pada <i>Frame</i> 60.....	125
Gambar 175. konten pada halaman “contoh hewan karnivora”	126
Gambar 176. penambahan informasi	126
Gambar 177. mengimpor <i>file</i> dengan nama berurutan	127
Gambar 178. menyesuaikan ulang posisi teks terhadap gambar	127
Gambar 179. menyesuaikan posisi <i>MovieClip</i> terhadap kontainer	127
Gambar 180. menambahkan informasi pada <i>Frame</i> 2 <i>MovieClip</i>	128
Gambar 181. penambahan informasi secara <i>Frame by Frame</i>	128
Gambar 182. penambahan <i>Linkage</i> dan hasil penambahan efek “ <i>fade</i> ”	130
Gambar 183. persiapan <i>file</i> suara.....	132
Gambar 184. menambahkan <i>Linkage</i> pada suara	132
Gambar 185. menambahkan <i>onOff</i> suara	134
Gambar 186. struktur <i>MovieClip onOff</i> suara.....	135
Gambar 187. menambahkan tombol “ <i>btn_setting</i> ”	135
Gambar 188. struktur <i>Timeline</i> pada <i>Frame</i> 15	136
Gambar 189. aset visual pada halaman <i>setting</i>	136
Gambar 190. menu <i>import</i> video	138
Gambar 191. memilih <i>file</i> video	138
Gambar 192. pemilihan <i>skin</i>	139
Gambar 193. tampilan setelah proses impor video.....	139
Gambar 194. pengaturan <i>component parameter</i> video	140
Gambar 195. persiapan aset visual untuk intro	141
Gambar 196. posisi objek di <i>Frame</i> 1 <i>Layer</i> 1	142
Gambar 197. posisi <i>MovieClip loading_bar</i>	142
Gambar 198. posisi <i>MovieClip loading_bar</i> pada <i>Frame</i> 1	142
Gambar 199. struktur <i>Layer</i>	143
Gambar 200. struktur pada <i>Frame</i> 61 dan penempatan logo.....	143
Gambar 201. struktur <i>Timeline</i> efek <i>fade in fade out</i> logo.....	143

Gambar 202. contoh <i>Popup</i> dalam multimedia “Menenal Lapisan Bumi”	145
Gambar 203. menambahkan tombol “btn_info”	145
Gambar 204. <i>MovieClip</i> popupMC	146
Gambar 205. menambahkan <i>file</i> huruf (<i>Embed font</i>)	147
Gambar 206. hasil penambahan fitur <i>popup</i>	147
Gambar 207. menambahkan <i>MovieClip</i> loginMC	148
Gambar 208. fitur <i>drag and drop</i> pada multimedia interaktif “Bersih-bersih”	149
Gambar 209. struktur <i>Timeline</i> pada Frame 80	150
Gambar 210. struktur halaman permainan.....	150
Gambar 211. konsep permainan <i>drag and drop</i>	151
Gambar 212. persiapan aset visual.....	152
Gambar 213. struktur <i>MovieClip</i> hewanMC.....	152
Gambar 214. struktur <i>MovieClip</i> makananMC	153
Gambar 215. struktur <i>MovieClip</i> hasilMC.....	153
Gambar 216. objek pada <i>Frame</i> 90 Layer konten	154
Gambar 217. menambahkan kuisMC.....	157
Gambar 218. <i>file</i> hewan yang akan diimpor sebagai gambar soal kuis	157
Gambar 219. stuktur halaman kuis	157
Gambar 220. hasil penambahan fitur kuis	160
Gambar 221. <i>file</i> gambar untuk <i>puzzle</i>	161
Gambar 222. menambahkan elemen <i>puzzle</i>	162
Gambar 223. menambahkan <i>Linkage</i> pada <i>Bitmap</i>	163
Gambar 224. elemen penyusun halaman <i>puzzle</i>	163
Gambar 225. tampilan fitur <i>puzzle</i>	164
Gambar 226. <i>MovieClip</i> timerMC	164
Gambar 227. menambahkan <i>MovieClip</i> timerMC	165
Gambar 228. hasil penambahan fitur <i>timer</i>	165
Gambar 229. <i>MovieClip</i> kursorMC.....	166
Gambar 230. Laboratorium Virtual berbasis HTML 5 Canvas	167
Gambar 231. struktur <i>Timeline</i> dan desain halaman judul.....	168
Gambar 232. konten <i>Frame</i> 20	169
Gambar 233. percobaan perubahan suhu dan tabel pengamatan.....	169
Gambar 234. <i>symbol</i> untuk membentuk simulasi	171
Gambar 235. Hasil simulasi.....	171
Gambar 236. penambahan <i>slider</i>	172
Gambar 237. hasil penambahan efek gravitasi.....	173
Gambar 238. menambahkan <i>rollmenu</i>	174
Gambar 239. hasil penambahan <i>rollMenu</i>	175
Gambar 240. hasil simulasi dengan rumus.....	176
Gambar 241. beberapa simulasi dalam <i>file MPI Component</i>	177
Gambar 242. pengaturan publikasi <i>desktop</i>	181

Gambar 243. proses <i>publishing</i> dan <i>file</i> yang dihasilkan	181
Gambar 244. halaman persetujuan Adobe Air SDK.....	182
Gambar 245. mengekstrak <i>file</i> AIR SDK	183
Gambar 246. menambahkan AIR SDK ke dalam Adobe Animate	183
Gambar 247. memilih target AIR for Android.....	184
Gambar 248. pengaturan panel <i>General</i>	184
Gambar 249. membuat dokumen <i>certificate</i>	185
Gambar 250. pengaturan <i>certificate</i>	185
Gambar 251. persiapan <i>file icon</i>	185
Gambar 252. pengaturan <i>Icon</i>	186
Gambar 253. proses publikasi AIR for Android.....	186
Gambar 254. mengatur perizinan instalasi aplikasi.....	187
Gambar 255. memilih <i>file</i> APK	187
Gambar 256. tampilan pada <i>mobile phone</i> Android	188
Gambar 257. mengubah ukuran dokumen	188
Gambar 258. pengaturan ulang aset visual setelah perubahan ukuran <i>Stage</i>	189
Gambar 259. membuka <i>file</i> –app.XML	189
Gambar 260. halaman <i>web</i> Android Studio	191
Gambar 261. menjalankan aplikasi Android Studio	191
Gambar 262. halaman <i>Setting</i>	192
Gambar 263. lokasi Android SDK	192
Gambar 264. lokasi file <i>adt.cfg</i>	193
Gambar 265. <i>rename file</i> APK.....	194
Gambar 266. tampilan multimedia interaktif pada <i>web browser</i>	195

DAFTAR TABEL

Tabel 7.1 List suara yang dibutuhkan	131
--	-----

BAB I

Konsep Dasar Multimedia Interaktif

1.1 Definisi

Istilah multimedia digunakan untuk mendefinisikan area yang sangat luas yang mencakup bidang informatika, telekomunikasi, sektor produksi audio visual, sinema dan digital media. Luasnya area multimedia tersebut mendorong munculnya istilah yang lebih spesifik, salah satunya adalah multimedia interaktif. Istilah multimedia interaktif menjadi sebuah istilah umum yang sering didengar di berbagai bidang dan secara khusus lebih banyak dibahas di ranah pendidikan dengan istilah yang lebih lengkap yaitu multimedia pembelajaran interaktif.

Apabila ditinjau secara etimologis, maka kata multimedia interaktif tersusun dari tiga kata, yaitu “multi” yang dalam bahasa Latin “*multus*” berarti banyak, media yang berasal dari kata “*medium*” yang artinya perantara, “interaktif” yang berasal dari kata “*interAction*” yang berarti hubungan antara dua hal yang saling mempengaruhi. Maka multimedia interaktif secara sederhana dapat didefinisikan sebagai perpaduan antara dua atau lebih media yang dapat dimanipulasi oleh penggunaannya. Dalam definisi ini segala kombinasi beberapa media yang melibatkan interaktivitas penggunaannya dapat disebut sebagai multimedia interaktif, tidak terikat dalam bentuk fisik atau digital. Sebagai contoh ketika beberapa orang bermain permainan papan seperti monopoli atau ular tangga. Permainan tersebut masuk ke dalam definisi multimedia pada tataran ini, karena di dalamnya terdapat beberapa media, seperti papan yang secara visual menampilkan informasi tertentu, dadu yang menunjukkan angka, kartu-kartu khusus, serta seperangkat aturan yang mengatur interaktivitas antar media dan pemain.

Ketika multimedia mengacu pada seni atau sistem pendidikan maka tersirat bahwa multimedia “menggunakan lebih dari satu media ekspresi atau komunikasi”. Interpretasi kata “ekspresi” dan “komunikasi” yang digunakan dalam definisi tersebut secara implisit menandakan keberadaan dari proses interaktif. Komunikasi dalam hal ini dapat dianggap sebagai proses interaksi antara dua pihak yang saling bertukar informasi, berkembang atau berubah sebagai hasilnya. Multimedia menjadi sebuah produk atau karya kreatif yang mendukung ekspresi atau komunikasi melalui berbagai media dengan kemampuan untuk mempengaruhi dan mengubah konten maupun konteksnya.

Dalam perkembangannya multimedia interaktif dikaitkan dengan teknologi komputer yang mengacu pada pemanfaatan komputer untuk membuat dan menggabungkan beberapa media berbasis teks, grafik, audio, gambar bergerak (video dan animasi) serta menggabungkan *link*

dan fitur-fitur tertentu yang memungkinkan penggunaannya untuk melakukan navigasi dan interaksi.

1.2 Komponen Multimedia Interaktif

Secara umum komponen penyusun multimedia interaktif meliputi antarmuka (*user interface*) yang di dalamnya tersusun atas konten (teks, grafik, audio, video atau animasi) serta interaktivitas yang diatur oleh seperangkat aturan.

1. Teks
Teks dalam multimedia merupakan kombinasi kalimat yang bertujuan untuk menjelaskan atau memberi keterangan terhadap suatu materi.
2. Grafik
Penggunaan grafik atau gambar dalam multimedia bertujuan untuk memperjelas materi yang ingin disampaikan.
3. Audio
Audio diartikan sebagai bunyi berbentuk digital seperti musik, suara, narasi dan lain-lain. Dalam multimedia audio akan membantu penekanan informasi yang ditampilkan.
4. Video
Video merupakan sarana penyampaian informasi yang memiliki kelebihan yaitu menarik, langsung dan efektif.
5. Animasi
Animasi dapat diartikan penggabungan media teks, gambar dan suara dalam satu set pergerakan. Animasi berguna untuk memvisualisasikan sesuatu selain dengan menggunakan video.
6. Interaktivitas
Elemen interaktivitas merupakan elemen utama di dalam sebuah multimedia interaktif. Beberapa aspek interaktif dapat berupa navigasi, permainan dan latihan. Interaktivitas diatur oleh seperangkat aturan khusus yang ditetapkan melalui tahapan pemrograman.

1.3 Metode Pengembangan Multimedia Interaktif

Dalam pengembangan multimedia interaktif dapat mengadaptasi dari berbagai macam metode. Metode yang paling umum dalam sebuah pengembangan aplikasi antara lain adalah *RnD* dan *SDLC* (Metode pengembangan *software* seperti metode *waterfall*, *agile*, *rapid*, *dynamic*, *spiral*, *feature driven* dan sebagainya). Pada buku ini akan digunakan penyederhanaan dari beberapa metode tersebut.

RnD atau *Research and Development* mengacu pada Borg dan Gall (1983) merupakan sebuah metode pengembangan produk yang di dalamnya terdapat proses pengembangan dan proses evaluasi, memiliki 10 tahapan yaitu : *research and information collecting*, *planning*, *develop preliminary form of product*, *preliminary field testing*, *main product revision*, *main field testing*,

operational product revision, operational field testing, final product revision, and dissemination and implementation.

1. *Research and information collecting*
merupakan tahapan awal dari suatu proses pengembangan produk, yaitu diawali dengan studi literatur yang berkaitan dengan permasalahan yang dikaji, dan persiapan untuk merumuskan kerangka kerja.
2. *Planning*
termasuk dalam langkah ini merumuskan kecakapan dan keahlian yang berkaitan dengan permasalahan, menentukan tujuan yang akan dicapai pada setiap tahapan, dan jika mungkin/diperlukan melaksanakan studi kelayakan secara terbatas;
3. *Develop preliminary form of product*
merupakan tahapan mengembangkan produk purwarupa atau prototipe dari produk yang akan dihasilkan. Produk yang dihasilkan pada tahapan ini sudah dapat dioperasikan secara penuh. Termasuk dalam langkah ini adalah persiapan komponen pendukung, menyiapkan pedoman dan buku petunjuk, dan melakukan evaluasi terhadap kelayakan alat-alat pendukung;
4. *Preliminary field testing*
merupakan tahapan ujicoba lapangan awal dalam skala terbatas, dengan melibatkan subjek sebanyak 6 – 12 subjek. Pada langkah ini pengumpulan dan analisis data dapat dilakukan dengan cara wawancara, observasi atau angket untuk mengetahui tingkat kelayakan produk dan kinerja fitur utama yang dimiliki produk.
5. *Main product revision*
yaitu melakukan perbaikan terhadap produk awal yang dihasilkan berdasarkan hasil ujicoba awal. Perbaikan ini sangat mungkin dilakukan lebih dari satu kali, sesuai dengan hasil yang ditunjukkan dalam ujicoba terbatas, sehingga diperoleh *draft* produk (model) utama yang siap diujicoba lebih luas;
6. *Main field testing*
merupakan uji coba utama yang melibatkan subjek yang lebih luas. Pada tahapan ini respon pengguna dikaji lebih detail terkait dengan fitur-fitur pendukung.
7. *Operational product revision*
merupakan tahapan perbaikan/penyempurnaan terhadap hasil uji coba lebih luas, sehingga produk yang dikembangkan menjadi desain model operasional yang siap divalidasi;
8. *Operational field testing*
yaitu langkah uji validasi terhadap model operasional yang telah dihasilkan, sekaligus pengujian penerimaan pasar. Pada tahapan ini seluruh komponen diuji kinerjanya, sekaligus diukur penerimaannya terhadap pasar.
9. *Final product revision*
yaitu melakukan perbaikan akhir terhadap model yang dikembangkan guna menghasilkan produk akhir (final);

10. *Dissemination and implementation*

yaitu langkah menyebarluaskan produk/model yang dikembangkan melalui target *platform* yang direncanakan.

Metode *SDLC* (*Systems Development Life Cycle*) merupakan sebuah siklus yang digunakan dalam pembuatan atau pengembangan sistem informasi yang bertujuan untuk menyelesaikan masalah secara efektif. Pada metode *SDLC* disusun sebuah tahapan kerja yang bertujuan untuk menghasilkan sistem berkualitas tinggi yang sesuai dengan keinginan pelanggan atau tujuan dibuatnya sistem tersebut. *SDLC* menjadi kerangka yang berisi langkah-langkah yang harus dilakukan untuk memproses pengembangan suatu perangkat lunak. Sistem ini berisi rencana lengkap untuk mengembangkan, memelihara, dan menggantikan perangkat lunak tertentu. Secara umum *SDLC* memiliki pola yang diambil untuk mengembangkan sistem perangkat lunak, yang terdiri dari tahap-tahap: rencana (*planning*), analisis (*analysis*), desain (*design*), implementasi (*implementation*), uji coba (*testing*) dan pengelolaan (*maintenance*).

1. *Perencanaan Sistem (Systems Planning)*

seperti halnya metode *RnD*, tahapan awal *SDLC* dimulai dari aspek studi kelayakan pengembangan sistem (*feasibility study*). Aktivitas-aktivitas yang ada meliputi :

- Pembentukan dan konsolidasi tim pengembang.
- Mendefinisikan tujuan dan ruang lingkup pengembangan.
- Mengidentifikasi apakah masalah-masalah yang ada bisa diselesaikan melalui pengembangan sistem.
- Menentukan dan evaluasi strategi yang akan digunakan dalam pengembangan sistem.
- Penentuan prioritas teknologi dan pemilihan aplikasi.

2. *Analisis Sistem (Systems Analysis)*

pada tahap ini, sistem akan dianalisis lebih detail mengenai kelebihan dan kekurangan sistem, fungsi sistem, pembaharuan yang dapat diterapkan sampai dengan unique selling yang mEmbedajan dengan kompetitor. Pada tahapan ini juga dianalisis terkait alokasi sumber daya, perencanaan kapasitas sumber daya yang dimiliki, penjadwalan, dan estimasi biaya, dan penetapan.

Analisa sistem adalah tahap di mana dilakukan beberapa aktivitas berikut:

- *Brainstorming* dalam tim pengembang terkait dengan produk yang akan dikembangkan.
- Mengklasifikasikan masalah, peluang, dan solusi yang mungkin diterapkan.
- Menganalisis produk kompetitor dan menyusun unique selling produk yang dikembangkan.
- Analisa kebutuhan pada sistem dan membuat batasan-batasan sistem.
- Mendefinisikan kebutuhan sistem.

- Mengatur jadwal pengembangan dan pembagian sumber daya.
3. *Perancangan Sistem (Systems Design)*
 tahapan ini akan menghasilkan prototipe dan beberapa output lain meliputi dokumen berisi desain, spesifikasi produk, fitur produk, dan komponen yang diperlukan untuk mewujudkan proyek tersebut. Pada tahap ini, aktivitas-aktivitas yang dilakukan adalah:
 - Menyusun sistem navigasi aplikasi.
 - Menyusun pola interaksi obyek dan fungsi pada masing-masing halaman navigasi.
 - Menyusun *flowchart* aplikasi.
 - Merancang *user interface*.
 4. *Implementasi Sistem (Systems Implementation)*
 tahap berikutnya adalah implementasi yaitu mengimplementasikan rancangan dari tahap-tahap sebelumnya dan melakukan uji coba. Dalam implementasi, dilakukan aktivitas-aktivitas sebagai berikut:
 - Pembuatan asset visual yang akan digunakan.
 - Pemrograman aplikasi berdasarkan desain sistem.
 - Pengujian dan perbaikan aplikasi (*debugging*).
 5. *Pemeliharaan Sistem (Systems Maintenance)*
 merupakan sebuah tahapan untuk menjaga sistem tetap mampu beroperasi secara benar. Pengembang harus responsif terhadap perubahan kebutuhan pengguna dan melakukan adaptasi atau perbaikan terkait kebutuhan tersebut.

Mengacu kepada 2 metode tersebut di atas dan kaitannya dengan pengembangan multimedia interaktif, buku ini menawarkan model pengembangan aplikasi yang menggabungkan metode *RnD* dan *SDLC* menjadi 5 tahapan, yaitu : *Design Treatment*, *Visual Development Phase*, *Programming Phase*, *Testing Phase* dan *Publishing and Maintenance Phase*.

1. *Design Treatment*
Design treatment merupakan sebuah istilah desain yang mengandung pengertian analisis tentang produk apa yang akan dibuat dan apa yang dibutuhkan untuk membuatnya. Pada tahapan ini dilakukan riset pendahuluan terkait produk yang dikembangkan, serta mendokumentasikan rencana atau tahapan yang akan dilakukan dalam pengembangan produk.
2. *Visual Development Phase*
 Tahapan ini meliputi tahapan menyusun antar muka aplikasi dan pengembangan aset visual yang dibutuhkan aplikasi.

3. *Programming Phase*

Pada tahapan ini dilakukan pemrograman fungsi utama dan fungsi tambahan dari aplikasi, serta dilakukan ujicoba internal untuk memastikan fitur utama aplikasi berjalan dengan baik.

4. *Testing Phase*

Testing phase dilakukan dengan cara mengujicoba produk dalam lingkungan terbatas untuk mendapatkan feedback dan ujicoba dalam lingkungan yang sebenarnya untuk mendapatkan respon pengguna terkait produk.

5. *Publishing and Maintenance Phase*

Pada tahapan ini produk didistribusikan sesuai dengan target *platform* yang direncanakan. Proses review terhadap produk terus dilakukan untuk menambahkan pembaharuan pada aplikasi.

1.4 Tujuan dan Target Buku

Buku ini secara khusus ditulis untuk melengkapi buku sebelumnya ([Desain dan Pemrograman Multimedia Interaktif](#)). Buku tersebut telah menjelaskan secara detail proses pengembangan multimedia pembelajaran interaktif dengan aplikasi Adobe Flash/Adobe Animate. Sementara buku ini lebih berfokus pada proses pengembangan multimedia interaktif yang lebih sederhana dengan memanfaatkan *precompiled clip* atau kode yang telah terkompilasi dengan penguatan konsep desain.

Buku ditujukan untuk pembelajar tingkat pemula yang ingin mengembangkan aplikasi multimedia dengan cara yang sederhana, pemrograman yang tidak kompleks dan hasil yang optimal untuk berbagai luaran *platform* seperti PC maupun Android. Buku ini memungkinkan untuk dipelajari di tingkat SMP, SMA, tingkat Universitas maupun untuk guru pengembang aplikasi pembelajaran.

1.5 Aplikasi dan Contoh Multimedia Interaktif

Dalam buku ini digunakan aplikasi Adobe Animate. Pembaca dapat menggunakan Adobe Animate versi 2017 ke atas, atau bagi pengguna Adobe Flash dapat menggunakan Adobe Flash versi CC 2014 ke atas. Dalam tutorial di buku ini digunakan Adobe Animate 2021 dengan ruang kerja model *classic* agar pembaca yang menggunakan aplikasi Adobe Animate lama atau Adobe Flash tetap dapat mengikuti tutorial yang ada.

Di dalam buku ini terdapat beberapa contoh aplikasi multimedia interaktif yang dibuat oleh mahasiswa Desain Komunikasi Visual Universitas Negeri Semarang, yang secara spesifik menyelesaikan karya multimedia interaktif sebagai bagian dari penyelesaian mata kuliah yang diampu oleh penulis. Penulis sedapat mungkin menampilkan *credit* karya mahasiswa yang ditampilkan di dalam buku ini.

Seluruh tutorial dan penjelasan di dalam buku ini terintegrasi dengan tutorial yang ada pada situs www.wandah.org dan penjelasan pada *channel* Youtube www.youtube.com/wandahw

1.6 Distribusi Buku

Buku ini diterbitkan secara online dalam bentuk *e-book* dan diedarkan secara gratis, dengan beberapa ketentuan sebagai berikut :

1. Pembaca diperbolehkan menyebarkan buku tanpa mengubah, menambah atau mengurangi konten buku.
2. Pembaca diperbolehkan mengutip atau mensitasi buku sesuai dengan kaidah yang berlaku.
3. Pembaca diperbolehkan mencetak buku untuk keperluan pribadi, namun tidak diperbolehkan mencetak untuk diperjual belikan (dilarang mengomersialkan buku ini dalam bentuk apapun).
4. Penambahan atribut nama penulis (*credit*) diperlukan pada setiap aplikasi yang dihasilkan.
5. Pembaca diperbolehkan menggunakan dan memodifikasi *file* tutorial untuk keperluan apapun, termasuk keperluan komersial.
6. Seluruh *file* sumber asli terdapat di situs www.wandah.org



BAB 2

Design Treatment

2.1 Multimedia sebagai bagian dari desain

Mengacu SKKNI desain grafis tahun 2016, multimedia interaktif termasuk ke dalam peta profesi di bidang desain grafis/Desain Komunikasi Visual (DKV). Desain Grafis/Desain Komunikasi Visual berdasarkan tujuannya merupakan bidang yang mencakup 3 (tiga) fungsi pentingnya yaitu *to inform*, *to indentify*, dan *to persuade*. Fungsi memberikan informasi, mengidentifikasi dan membujuk (mempersuasi) dengan mempergunakan media grafis (berbasis cetak), media digital, dan *environmental* media (lingkungan). DKV sebagai bidang ilmu multidisiplin yang luas melingkupi berbagai bidang terkait termasuk di dalamnya penggunaan media audio visual untuk mencapai tujuan komunikasi. Adapun lingkup profesi yang luas tersebut terbagi menjadi profesi yang lebih umum biasanya disebut desainer grafis/komunikasi visual dan profesi yang lebih spesialis seperti: desainer *brand*, desainer kemasan, desainer *website*, desainer multimedia, dan lain-lain.



Gambar 1. peta profesi DKV dengan multimedia interaktif berada di dalamnya
(sumber: SKKNI Desain Grafis/DKV, 2016)

Sangat wajar ketika multimedia interaktif menjadi subbagian dari desain grafis/Desain Komunikasi Visual karena multimedia sendiri sangat intens dengan keberadaan grafis sebagai media utama. Tujuan utama multimedia interaktif sejalan dengan tujuan utama Desain Komunikasi Visual. Oleh karena itu, dalam tahapan perencanaan atau *design treatment* ilmu pengetahuan terkait desain sangat mutlak diperlukan.

Dalam tahapan awal mengembangkan multimedia interaktif perlu ditanamkan konsep dasar bahwa multimedia adalah bagian dari desain yang bertujuan sebagai pengantar pesan/informasi. Konsep ini menitikberatkan akan pentingnya aspek informatif sehingga membutuhkan wawasan mengenai teori komunikasi untuk menjabarkan informasi dalam format audio visual yang interaktif. Hal tersebut menyangkut beberapa hal yaitu Pesan/*message* (apa yang akan diinformasikan); Khalayak/*audience* (siapa khalayak yang dituju); dan Sasaran/*objective* (apa tujuan yang diharapkan).

Multimedia sebagai bagian dari bidang desain menuntut beberapa pengetahuan dasar kesenirupaan umum dan keterampilan/kepekaan khusus, diantaranya adalah:

1. Pengetahuan, keterampilan dan kepekaan mengenai unsur desain (titik, garis, bidang, ruang, tekstur, dan warna) dan prinsip desain (kesatuan, keserasian, irama, penekanan, keseimbangan, dan proporsi).
2. Memiliki pengetahuan dan keterampilan dalam grafis dasar.
3. Memiliki pengetahuan dasar animasi/*motion Graphic*.
4. Memiliki pengetahuan dasar audio visual.
5. Memiliki pengetahuan dasar pemrograman.

2.2 Unsur Desain

Dalam merencanakan sebuah produk multimedia interaktif diperlukan pengetahuan dasar kesenirupaan untuk menghasilkan tampilan visual yang menarik dan menyajikan makna yang tepat. Dalam seni rupa, dikenal adanya unsur-unsur seni rupa atau ketika berbicara di ranah desain, maka dapat disebut sebagai unsur-unsur desain. Beberapa studi literatur memiliki variasi dalam menjabarkan tentang unsur-unsur desain, namun secara umum dapat diperoleh kesamaan bahwa unsur desain terdiri dari :

1. Titik
titik adalah satu bentuk kecil tanpa dimensi. Pada umumnya titik ditampilkan dalam bentuk bundaran kecil, sederhana, mampat, tanpasudut, dan tanpa arah. Titik cenderung ditampilkan dalam bentuk kelompok, dengan variasi jumlah, susunan, dan kepadatan tertentu.
2. Garis (*Line*)
garis merupakan gabungan dari satu titik dengan titik yang lain yang membentuk suatu objek. Bentuk garis dapat berupa garis vertikal, horizontal, diagonal, kurva, putus-putus, zig-zag, dan tidak beraturan untuk menghasilkan makna tertentu.

3. Raut/Bidang/Bentuk (*Shape*)
bidang merupakan garis yang ujungnya saling bertemu dan membuat area tertutup. Bentuk-bentuk ini dapat berupa bentuk dasar seperti kotak, lingkaran, segitiga, lonjong, maupun yang lebih kompleks seperti bentuk geometris atau yang dapat diukur, bentuk natural, dan bentuk abstrak.
4. Tekstur (*Texture*)
tekstur merupakan permukaan dari suatu benda yang dapat dirasakan dengan cara diraba atau dilihat, maupun gabungan dari keduanya. Tekstur dalam kehidupan nyata dapat berupa permukaan pasir, tekstil, permukaan batuan dan lain-lain. Tekstur terbagi menjadi dua, yaitu tekstur nyata (dapat disentuh dan dilihat) dan tekstur semu (hanya dapat dilihat/visual). Tekstur dapat dalam desain dapat digunakan untuk memberikan pengalaman berbeda pada saat dilihat.
5. Kontras (*Contrast*)
Kontras merupakan warna yang berlawanan. Dalam kontras terdapat perbedaan warna antara warna yang satu dengan warna yang lain. Contoh warna kontras adalah merah dengan hijau, kuning dengan ungu, dan biru dengan oranye. Dalam desain, kontras digunakan untuk menonjolkan pesan ataupun informasi yang disampaikan. Adanya kontras membantu meningkatkan keterbacaan, fokus, dan titik berat dalam desain.
6. Ukuran (*Size*)
Ukuran merupakan besar kecilnya objek visual. Ukuran visual dapat menciptakan penekanan pada objek visual. Penentuan ukuran visual dari sebuah objek visual menentukan hirarki dari suatu desain. Dengan penggunaan objek visual audiens dapat mengetahui objek mana yang pertama kali dilihat atau dibaca sehingga pesan atau informasi yang ingin disampaikan kepada audiens akan lebih mudah untuk dipahami.
7. Warna (*Color*)
Warna dapat diartikan sebagai corak rupa atau kesan yang ditangkap mata dari cahaya yang dipantulkan oleh suatu benda. Warna dapat berfungsi sebagai identitas dan penyampai pesan, hal ini dikarenakan setiap warna memiliki karakter dan sifat yang berbeda-beda. Dalam sebuah desain, desainer perlu memperhatikan kesan apa yang ingin disampaikan melalui warna tersebut.
8. Ruang (*Space*)
Ruang merupakan jarak yang memisahkan sesuatu atau area yang terdapat di sekitar objek yang digunakan untuk memisahkan unsur-unsur desain lainnya. Ruang atau space ini berfungsi untuk membedakan antara unsur desain yang satu dengan lainnya. Dalam penerapannya, ruang sangat membantu dalam proses layouting atau penataan elemen-elemen desain. Dengan adanya ruang audiens akan lebih mudah untuk membedakan judul, paragraf, kapan harus membaca dan kapan harus berhenti. Selain itu, ruang juga berfungsi sebagai tempat istirahat mata agar mata tidak lelah saat membaca informasi dalam sebuah desain.

2.3 Prinsip Desain

Prinsip-prinsip desain merupakan kerangka dasar dalam menggabungkan elemen-elemen/unsur-unsur desain agar menghasilkan komunikasi yang efektif atau menciptakan interaksi antar elemen dalam desain. Dengan memahami prinsip desain yang baik, maka seorang desainer aplikasi akan dapat menghadirkan sebuah tampilan visual yang secara universal mudah untuk diterima dan dinyatakan baik. Beberapa literatur desain menjabarkan prinsip-prinsip desain sebagai berikut :

1. Prinsip Kesatuan (*Unity*)
prinsip kesatuan dalam desain bertujuan untuk menciptakan citra yang terintegrasi, di mana semua elemen bekerja sama untuk mendukung desain secara keseluruhan. Dalam multimedia interaktif, kesatuan dibentuk dengan adanya keseragaman antara desain antar muka pada masing-masing halaman. Keseragaman dapat dibentuk melalui beberapa elemen seperti penggunaan warna, penggunaan tipografi, tata letak yang konsisten, konsistensi pengayaan gambar, serta penggunaan elemen yang berkelanjutan antara halaman satu dengan halaman lainnya.
2. Prinsip Keserasian (*Harmony*)
prinsip harmoni bertujuan untuk menciptakan keteraturan tatanan diantara bagian masing-masing halaman multimedia. Prinsip harmoni berfokus untuk membentuk suatu pola yang memenuhi kaidah-kaidah estetik serta mengutamakan aspek keselarasan dan kepantasan. Harmoni dapat dibentuk melalui tata letak yang teratur, konsisten serta penggunaan warna yang selaras dengan materi yang ditampilkan di dalam multimedia.
3. Prinsip Irama (*Rhythm*)
ritme merupakan prinsip desain yang muncul akibat adanya pengulangan pada bidang/ruang yang menghasilkan gerakan, getaran, atau perpindahan semu dari unsur satu ke unsur lain. Ketika terjadi pengulangan yang dinamis dan mengalir, maka akan terbentuk sebuah aliran (*eyeflow*). Dalam multimedia prinsip ritme dapat ditampilkan pada sebuah halaman dengan menampilkan perulangan elemen desain dengan pengaturan jarak dan ukuran yang dinamis, selain itu ritme juga dapat ditampilkan melalui efek animasi yang dimunculkan pada sebuah halaman.
4. Prinsip Penekanan/Dominasi (*Emphasis*)
desain sebagai salah satu bentuk komunikasi selalu menekankan satu gagasan utama yang harus lebih mendominasi elemen-elemen lainnya. Hal tersebut bertujuan untuk mengarahkan pandangan penggunanya pada suatu yang ditonjolkan. Prinsip penekanan/emphasis dapat diwujudkan dengan mengganti ukuran menjadi lebih besar dalam sebuah area desain, menampilkan warna yang kontras, ukuran huruf dan bentuk yang menonjol serta mengatur irama/*eyeflow* dari sebuah halaman desain.
5. Prinsip Keseimbangan (*Balance*)
keseimbangan merupakan kesan yang diciptakan oleh pemerataan bobot visual di antara semua elemen desain. Keseimbangan visual terjadi ketika berat satu atau lebih

elemen desain didistribusikan secara merata atau proporsional dalam ruang. Keseimbangan yang dibentuk dapat bersifat simetris dan asimetri. Keseimbangan simetris, dibentuk dengan meletakkan elemen desain secara seimbang berdasarkan sebuah sumbu pusat. Keseimbangan simetris akan berusaha menyeimbangkan elemen seperti halnya pencerminan. Sementara keseimbangan asimetri disusun dengan menyeimbangkan satu elemen dengan bobot elemen yang berlawanan tanpa mencerminkan elemen di kedua sisi sumbu pusat.

6. Prinsip Proporsi (*Proportion*)

proporsi merupakan hubungan perbandingan antara bagian yang satu elemen dengan elemen yang lain, atau dengan elemen keseluruhan. Berbeda dengan prinsip emphasis, proporsi lebih menekankan relasi atau hubungan antar elemen yang sejajar derajatnya berdasarkan hirarki. Proporsi dibentuk oleh tampilan ukuran, warna, dan penempatan posisi elemen dalam desain.

2.3 Merencanakan Multimedia Interaktif

Dalam merencanakan sebuah proyek multimedia interaktif telah ditentukan beberapa tahapan seperti pada penjelasan sebelumnya. Tahap yang paling awal merupakan *design treatment* yang dapat dijabarkan prosesnya sebagai berikut :

1. Tahapan riset


pengembangan multimedia interaktif perlu diawali dengan riset pendahuluan terkait dengan produk yang akan dibuat. Riset atau penelitian dapat melibatkan metode yang bervariasi baik secara kuantitatif maupun kualitatif, dengan melibatkan teknik pengumpulan data yang bervariasi tergantung dengan kebutuhan. Beberapa permasalahan terkait dengan media yang akan dibuat beserta dengan referensi yang digunakan perlu dikaji secara mendalam untuk mendapatkan sudut pandang yang luas. Data yang diperoleh selanjutnya didokumentasikan dan dipilah menjadi beberapa bagian yang selanjutnya menjadi sistem navigasi utama dari multimedia interaktif.

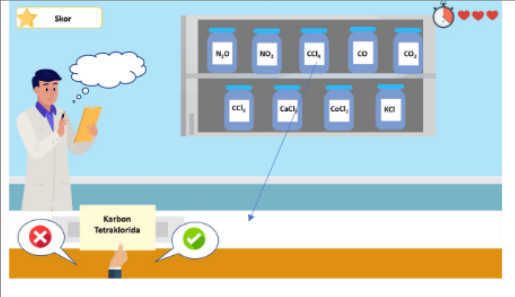
2. Menyusun *design document*/sketsa awal

pada tahapan berikutnya, data penelitian yang telah didapatkan ditampilkan dalam rancangan halaman aplikasi. Pada tahapan ini terdapat beberapa teknik yang dapat digunakan seperti melalui pembuatan dokumen desain yang spesifik dan teratur, atau cukup dengan membuat sketsa awal masing-masing halaman yang akan dibuat.

Sebagai contoh dokumentasi yang dibuat oleh Inayah Adi Oktaviana (2022) terkait dengan multimedia pembelajaran interaktif berjudul “Laboran”. Pada dokumentasi tersebut ditampilkan informasi penting terkait pembagian halaman, mockup tampilan visual yang diharapkan, keterangan tampilan, narasi audio yang digunakan, serta sistem interaktivitas yang ada pada halaman tersebut. Penyusunan dokumen desain yang detail seperti pada contoh ini akan mempermudah proses pengembangan aplikasi apabila dikerjakan oleh pihak ketiga atau oleh tim pengembang. Tim Pengembang dapat memahami dengan mudah struktur

navigasi dan fitur-fitur yang harus ditambahkan di masing-masing halaman karena telah dideskripsikan dengan sangat spesifik.

Judul : Halaman Utama Nama : Home	No Frame : 5 No Halaman:
	
Keterangan Tampilan Background Laboratorium dan Karakter Kimi. Tombol Story Mode Tombol Game Mode Tombol Informasi Tombol Info Pengembang Tombol Sound On/Off Tombol X untuk exit	Narasi Audio Background Music Hover Tombol
Role Game and Navigasi <ol style="list-style-type: none"> 1. Tombol Story Mode menuju kepada game dengan alur dan modul (Frame 8) 2. Tombol Game Mode menuju pada game dengan soal secara acak (Frame 54) 3. Tombol Petunjuk menuju ke halaman petunjuk permainan 4. Tombol Informasi berisi Kompetensi dan Indikator, kemudian informasi penulis, pengkaji media dan materi, dan pengembang, serta referensi 5. Tombol Sound on/off untuk menyalakan dan mematikan sound 6. Tombol X untuk exit 	

Judul : Story Level 1 Nama : Story Level 1	No Frame : 11 No Halaman:
	
Keterangan Tampilan Karakter Kimi Botol-botol senyawa yang bergerak dengan lambat Kertas Nama senyawa Simbol Benar dan Salah Skor Stopwatch 1 menit 3 Nyawa	Narasi Audio Background Music Sound suara benar Sound Suara salah
Role Game and Navigasi <ol style="list-style-type: none"> 1. Jika Botol yang diletakkan sesuai dengan request senyawa, maka skor +20 2. Jika Botol yang diletakkan benar, muncul simbol Benar 3. Jika Botol yang diletakkan salah, muncul simbol Salah 4. Jika botol salah, maka nyawa berkurang 1 5. Setelah botol di letakkan di meja, akan muncul request baru 6. Setiap request senyawa memiliki waktu tunggu 15 detik 7. Jika nyawa habis, point <100, menuju ke Frame Gagal Level 1 8. Jika point ≥ 100, menuju ke Frame Sukses Level 1 	

Gambar 2. dokumentasi multimedia pembelajaran interaktif (Oktaviana, 2022)

Contoh berikutnya adalah dengan menggunakan teknik sketsa, yaitu multimedia interaktif “Batik Kita” karya Zunalia Ahmad (2021). Konsep multimedia yang akan dibuat adalah sebagai media informasi terkait dengan Batik, yang meliputi definisi batik, jenis motif batik, proses pembuatan batik, dan penambahan permainan interaktif seperti kuis dan *puzzle* batik. Sistem dokumentasi yang digunakan berbeda dengan sistem dokumentasi tertulis seperti pada multimedia interaktif “laboran” di atas, namun digunakan konsep visual tiap-tiap halaman yang akan dibuat atau dalam desain disebut sebagai *rough layout*.



Gambar 3. rencana halaman awal multimedia interaktif “Batik Kita” (Ahmad, 2021)



Gambar 4. rencana sub halaman multimedia interaktif “Batik Kita” (Ahmad, 2021)

Konsep desain yang menarik ditampilkan oleh Aulia Rizka (2022) yang menyampaikan materi terkait pengenalan profesi pemadam kebakaran melalui multimedia interaktif dengan konsep *role play*. Pemain seolah-olah berperan sebagai Ben, seekor kucing pemadam kebakaran yang selanjutnya akan mendapatkan beberapa misi untuk memadamkan api sambil belajar tentang perangkat yang digunakan.





Gambar 5. rencana halaman multimedia interaktif “firefighter” (Rizka, 2022)

Penggunaan teknik sketsa dapat digunakan pada pengembangan aplikasi yang dilakukan secara personal, atau melibatkan tim kecil yang mudah untuk dikordinasikan. Pada teknik ini desain dan fitur masih tergambar secara kasar/tidak mendetail dan membutuhkan sentuhan personal (pengembang utama) untuk menambahkan detail-detail di tiap-tiap halaman multimedia.

2.4 Pengelolaan sumber daya dan perencanaan jadwal kerja

Setelah perencanaan dalam bentuk dokumen atau dalam bentuk sketsa selesai dibuat, pada tahapan berikutnya ditentukan sumber daya yang diperlukan dan pengaturan jadwal pengerjaan aplikasi. Dalam pengembangan aplikasi secara personal, tahapan ini dapat dilakukan dengan cara mengukur kapasitas pribadi mulai dari kemampuan diri dalam mengembangkan aset visual, *skill* pemrograman serta sumberdaya pendukung seperti sarana prasarana berupa *software* dan *hardware*. Sementara dalam pengembangan yang melibatkan tim, diperlukan pengaturan secara spesifik terkait kinerja masing-masing tim (atau sering diistilahkan dengan *task management*).

Manajemen tugas tim (*task management*) yang efektif memerlukan pengelolaan semua aspek tugas, termasuk status, prioritas, waktu, penugasan sumber daya manusia dan keuangan, pengulangan, ketergantungan, pemberitahuan, dan sebagainya. Hal tersebut berfungsi sebagai

dasar untuk alur kerja yang efisien dalam suatu tim pengembang. Mengelola tugas tim dapat saat ini dapat dibantu oleh perangkat lunak khusus yang banyak tersedia secara *online*. Kendati demikian terdapat beberapa prinsip dasar dalam pengelolaan tim dalam proyek multimedia yang harus diperhatikan agar berjalan efektif, yaitu :

1. manajer proyek telah menyusun dokumen desain yang spesifik dan dikomunikasikan secara menyeluruh ke tim pengembang, untuk mendapatkan kesepahaman.
2. tim pengembang aset visual harus bekerja lebih dahulu untuk menyediakan aset-aset visual utama yang akan digunakan dalam pengembangan navigasi dasar dan fitur utama/fitur inti aplikasi. Selanjutnya tim pengembang aset visual mengerjakan aset-aset pendukung aplikasi.
3. *programer* aplikasi bekerja setelah mendapatkan aset visual utama dan mengerjakan fitur utama terlebih dahulu.
4. fitur tambahan dikerjakan setelah fitur utama bekerja dengan baik.
5. masing-masing tim harus bekerja sesuai dengan *Timeline* yang ditetapkan, karena berimplikasi terhadap kinerja tim lainnya.

Melalui tahapan tersebut akan diperoleh gambaran waktu dan biaya yang diperlukan dalam pengembangan sebuah aplikasi multimedia. Sebagai contoh di bawah ini ditampilkan pembagian tugas dan *Timeline* pekerjaan per satuan minggu untuk mengerjakan sebuah proyek multimedia interaktif.

Detail Pekerjaan	Penanggung Jawab	Timeline (Agustus – September 2022)							
		Minggu 1 Agt	Minggu 2 Agt	Minggu 3 Agt	Minggu 4 Agt	Minggu 1 Sep	Minggu 2 Sep	Minggu 3 Sep	Minggu 4 Sep
Grafis UI : - Navigasi Utama	Tim Grafis								
Grafis Fitur Tambahan	Tim Grafis								
Coding Navigasi	Tim Programing								
Coding Fitur Tambahan	Tim Programing								
Audio	Tim Audio								
Uji Coba produk (beta)	Tester								
Revisi produk	Tim Grafis + Programming								
Produk Final	Tim Grafis + Programming								

Gambar 6. penjadwalan kerja tim pengembang aplikasi

Setelah tahapan penyusunan jadwal kerja dan pembagian sumber daya, maka tahapan pengembangan multimedia interaktif dapat dilanjutkan dengan tahapan pengembangan visual (*visual development phase*).

BAB 3

Visual Development Phase

Desain Antar Muka (*User Interface*)

Multimedia interaktif sebagai sebuah produk desain yang produk akhirnya akan digunakan oleh publik dapat mengadaptasi model AIDA. Model AIDA diadaptasi dari bidang pemasaran yang secara khusus bertujuan untuk meningkatkan perkembangan bisnis. AIDA merupakan singkatan dari *Attention* (Perhatian), *Interest* (Minat), *Desire* (Keinginan), dan *Action* (Aksi). Pada prinsip pertama yaitu *Attention* (perhatian) menuntut sebuah produk harus menarik perhatian. Dalam desain, menarik perhatian secara umum dapat dicapai dengan menampilkan visual yang “*eyecatching*”. Tahap *attention* merupakan tahap awal dimana konsumen atau pengguna mulai menyadari tampilan awal sebuah produk. Melalui desain atau tampilan visual yang menarik perhatian, maka tahapan AIDA berikutnya sampai tahapan *Action* (aksi) akan dapat dicapai.

Desain, khususnya desain grafis sebagai fundamental tampilan visual dari sebuah produk multimedia secara spesifik merupakan bidang ilmu yang kompleks, membutuhkan kepekaan estetis sekaligus keterampilan teknis dalam mewujudkannya. Pada tahapan pengembangan visual (*visual development phase*) sarat dengan kemampuan tim pengembang di bidang grafis.

Pada pengembangan multimedia interaktif secara personal, biasanya mengandalkan kapasitas personal dalam mengembangkan grafis atau mengandalkan sumber gambar digital yang tersedia di *website* stok grafis. Penggunaan sumber gambar digital yang berasal dari *web* stok grafis memungkinkan penggunaan grafis yang berulang, sehingga dimungkinkan akan mengalami kesamaan elemen visual antara produk yang dikembangkan dengan produk kompetitor. Sementara pengembangan oleh tim akan lebih eksklusif ketika menggunakan grafis yang diproduksi sendiri oleh tim, sehingga dipastikan memiliki tampilan yang berbeda dengan produk kompetitor. Mengembangkan grafis baik secara personal atau melalui tim, dapat mengacu pada beberapa prinsip desain serta pada konsep dasar desain antarmuka (*user interface*).

3.1 Desain Antarmuka (*User Interface*)

Desain antarmuka atau *user interface*, merupakan sebuah tampilan visual yang dilihat oleh pengguna dan digunakan untuk berinteraksi dengan sistem operasi aplikasi melalui elemen grafis yang ditampilkan. Dalam bidang pengembangan aplikasi terdapat dua jenis antar muka, yaitu *Command Line Interface (CLI)* dan *Graphical User Interface (GUI)*. Aplikasi multimedia interaktif masuk ke dalam jenis *Graphical user interface*, karena di dalamnya terdapat sebuah elemen-elemen grafis seperti ilustrasi, tombol, icon dan sebagainya.

Tujuan dari antarmuka adalah untuk membuat pengalaman pengguna aplikasi menjadi sederhana dan intuitif, membutuhkan sedikit usaha untuk mendapatkan hasil maksimal yang diinginkan. Dalam mewujudkannya diperlukan pemahaman terhadap prinsip desain secara umum (*unity, balance, harmony, rhythm, emphasis, proportion*) dan prinsip desain antar muka, yaitu :

1. Kejelasan (*Clarity*)

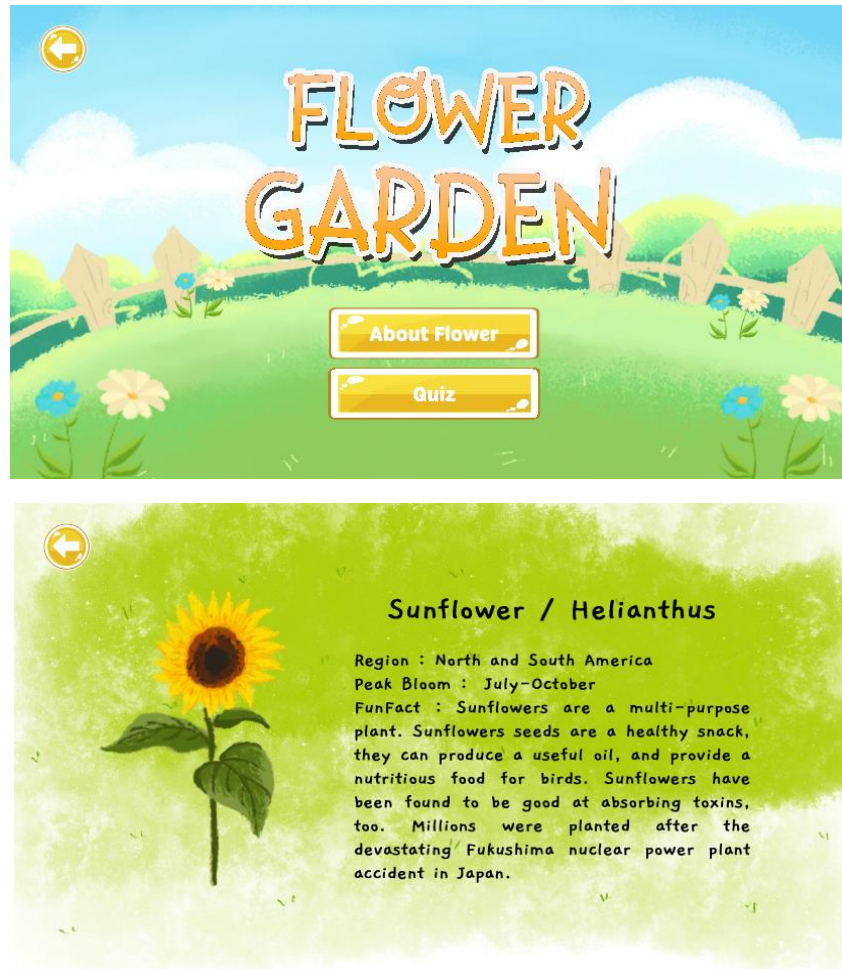
tampilan visual multimedia interaktif haruslah sederhana, singkat, padat, dan jelas. Tampilan layar aplikasi harus jelas, dalam artian mudah untuk dilihat mudah untuk dibaca, penggunaan warna yang mudah untuk dilihat oleh mata tanpa mengubah fokus mata terlalu sering. Huruf yang terbaca dengan baik dengan ukuran yang tepat.



Gambar 7. tampilan halaman “Vegefruits” (Arzak, 2021)

Sebagai contoh tampilan visual multimedia interaktif karya Naelur Arzak (2021) yang menampilkan tampilan visual yang memiliki kejelasan yang tinggi. Penggunaan warna huruf dengan latar yang baik mendukung keterbacaan masing-masing elemen. Penggunaan teks berwarna putih pada tombol ditambahkan efek bayangan (*shadow*

border), dan menggunakan warna latar yang lebih gelap untuk mendukung keterbacaan yang baik.



Gambar 8. tampilan halaman “Flower Garden” (Ulqalbi, 2021)

Contoh kedua multimedia interaktif tentang jenis tanaman berjudul “Flower Garden” karya Elsha Asyifa Ulqalbi (2021) pada dasarnya memiliki tampilan desain yang baik, namun pada beberapa bagian memiliki kejelasan yang masih kurang. Penggunaan teks dengan warna terang (putih) dengan latar kuning pada tombol memiliki kontras yang kurang baik, seharusnya teks berwarna putih perlu penambahan efek bayangan atau perlu penambahan *outline* berwarna gelap untuk meningkatkan keterbacaan. Pada contoh halaman berikutnya, penggunaan tekstur berwarna hijau dengan teks non-formal (bertipe *fancy*) mengurangi keterbacaan konten, sehingga perlu diubah.

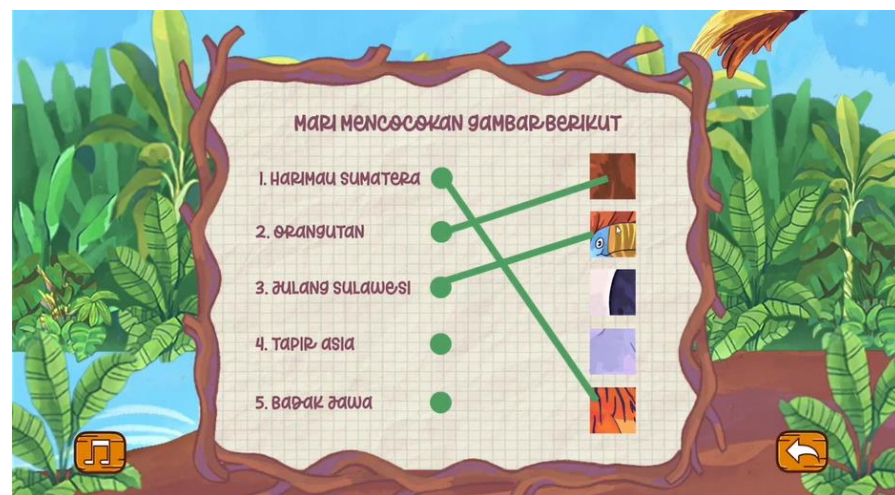
2. Konsistensi (*Consistency*)

konsistensi dapat dicapai dengan menentukan kesamaan antar halaman antarmuka. Kesamaan tata letak, kesamaan posisi, kesamaan penggunaan huruf dan tema warna.

Sebagai contoh, dalam desain terdapat konsep sederhana untuk menghasilkan konsistensi sebagai berikut :

- Tentukan warna tema, dan terapkan di masing-masing halaman
- Tentukan 3 jenis huruf, yaitu huruf untuk judul, huruf untuk subjudul dan huruf untuk *bodytext* (paragraf). Tentukan ukuran, warna huruf dan terapkan di masing-masing halaman.
- Tentukan posisi tombol yang konsisten, sebagai contoh tombol beranda (*home*) diletakkan di pojok kiri atas dan tombol kembali (*back*) di pojok kanan atas. Peletakan tombol ini harus konsisten di seluruh halaman.



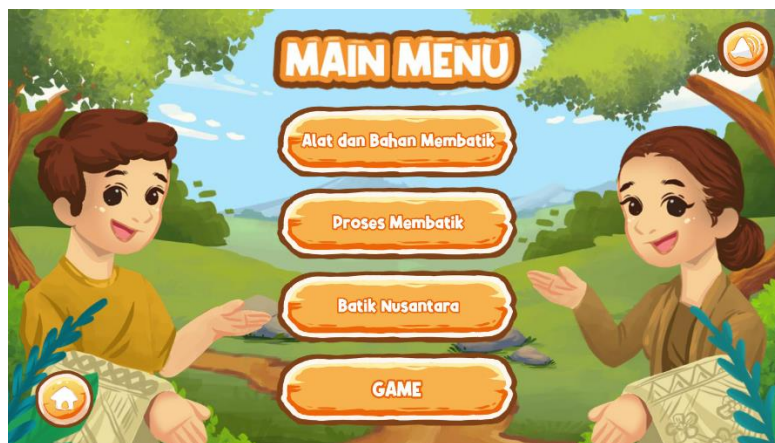


Gambar 9. multimedia “Satwa Endemik Indonesia” (Saad, 2021)

Pada contoh di atas, multimedia interaktif “Satwa Endemik Indonesia” karya Kamal Saad (2021) memiliki konsistensi yang baik dalam penggunaan aset visual, warna, huruf dan tata letak. Konsistensi tersebut akan menghasilkan pengalaman pengguna yang baik, sehingga terkait dengan prinsip lainnya.

3. Keakraban (*Familiarity*)

keakraban terkait dengan pengalaman pengguna, yaitu seberapa mudah pengguna menggunakan aplikasi karena memiliki kemiripan operasional dengan beberapa aplikasi populer yang telah ada sebelumnya. Tata letak mengikuti standar aplikasi populer agar mudah dikenali oleh pengguna.



Gambar 10. multimedia interaktif “Sinau Batik” (Amin, 2021)

Multimedia interaktif “Sinau Batik” karya Muhammad Hafizhuddin Amin (2021) memiliki tata letak yang relatif familiar dengan aplikasi-aplikasi populer, sebagai contoh aplikasi multimedia “Panduan Puasa Ramadhan” karya Educa Studio. Peletakan tombol *play* yang

berada tengah layar, dan tombol pengaturan di pojok atas atau bawah dapat ditemui hampir di sebagian besar aplikasi, sehingga menjadi elemen yang familiar bagi pengguna.

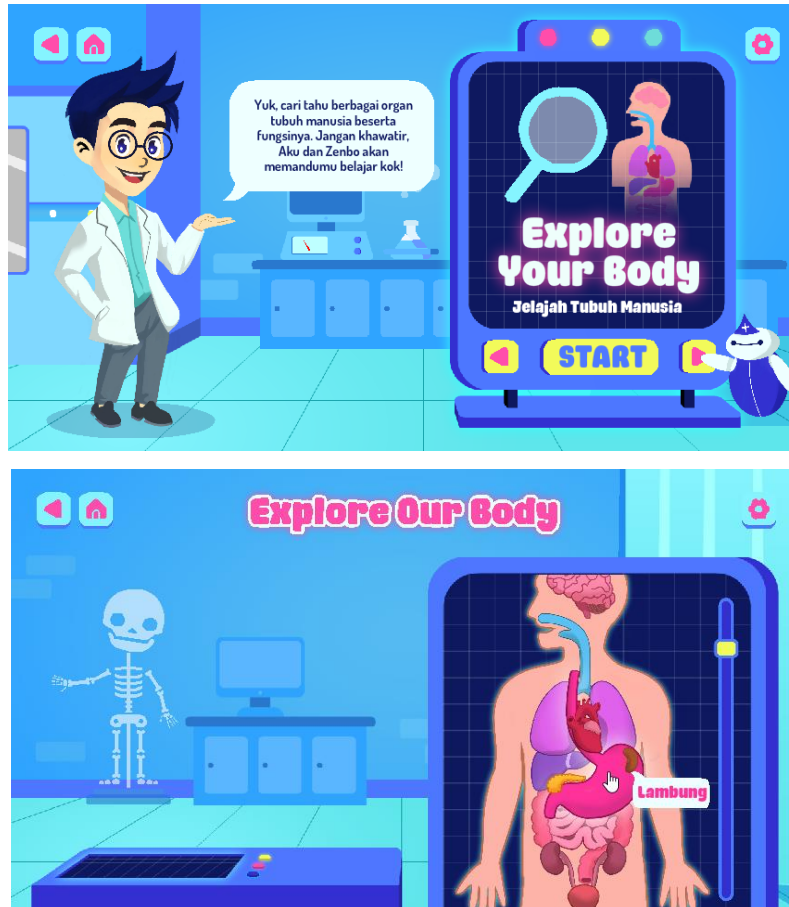


Gambar 11. aplikasi Marbel Panduan Puasa Ramadhan (Educa Studio)

4. Kontrol Pengguna (*Control*)

dalam merancang sebuah aplikasi, selalu diingat bahwa pengembang tidak membuat / merancang antarmuka untuk dirinya sendiri, melainkan untuk kebutuhan *user* yang beragam dengan berlatar belakang yang berbeda. Kemudahan dalam mengoperasikan aplikasi, kemudahan dalam memahami *icon*, *menu* dan elemen interaktif lainnya akan membentuk kurva belajar pengguna. Aplikasi yang baik memiliki kontrol pengguna yang mudah dipahami dan dapat dipelajari operasionalnya dengan cepat.

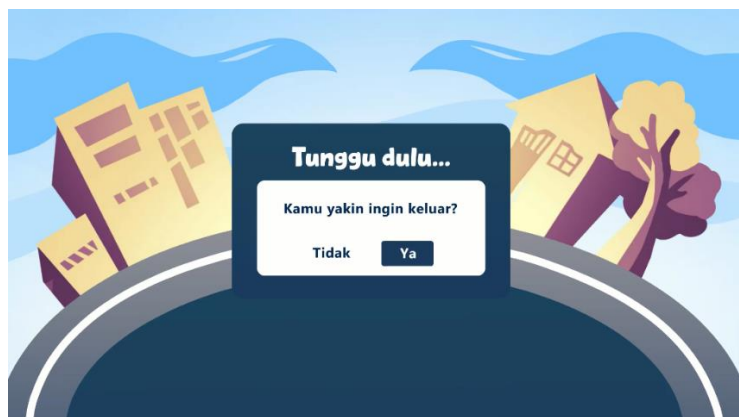
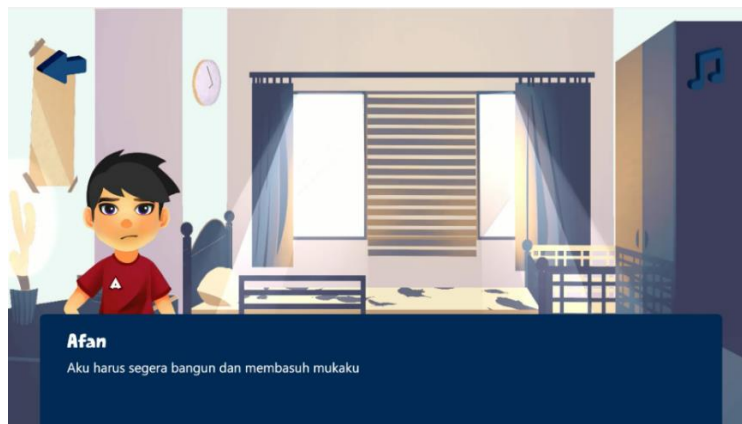




Gambar 12. multimedia “Inside Our Body” (Mustaqim, 2021)

Pada multimedia interaktif “Inside Our Body” karya Arzul Mustaqim (2021) memiliki sistem navigasi yang sangat kompleks untuk mengakses banyak fitur yang ada di dalamnya. Pada aplikasi tersebut, pengguna dipandu oleh beberapa fitur seperti tombol yang menyala/bersinar setelah beberapa saat, agen pedagogik yang menampilkan balon kata berisi petunjuk, atau label yang muncul ketika *mouse* melintas di atas objek (*mouse over*).

5. Umpan Balik (*Feedback*)
dalam konteks perancangan interaksi, *feedback* bertujuan untuk memberi tahu pengguna akan konsekuensi dari suatu tindakan. Pengguna menjadi sepenuhnya paham apa yang sedang terjadi dan mampu memperkirakan apa yang akan terjadi setelah melakukan tindakan. Contoh yang paling umum adalah peringatan ketika akan keluar dari aplikasi.



Gambar 13. aplikasi multimedia interaktif “Cerita Afan” (Wandayanti, 2022)

Pada aplikasi multimedia interaktif “Cerita Afan” karya Mila Wandayanti (2022) memiliki beberapa umpan balik ketika pengguna memilih sebuah menu tertentu. Menu utama ditampilkan dalam beberapa tombol berilustrasi, yang ketika ilustrasi ditekan maka akan muncul umpan balik yang memberikan instruksi kepada pengguna. Demikian halnya ketika akan keluar dari aplikasi, maka akan muncul dialog (*popup*) untuk memastikan apakah pengguna memang bermaksud keluar dari aplikasi.

6. Responsif (*Responsive*)

prinsip responsif dalam sebuah aplikasi dapat mengacu pada dua hal, pertama adalah kecepatan aplikasi dalam menanggapi sebuah *input* dari pengguna. Sebuah aplikasi membutuhkan sumber daya perangkat keras seperti *prosesor*, memori dan memori grafis untuk menjalankan fitur-fitur di dalamnya. Semakin kompleks sebuah aplikasi dan semakin tinggi spesifikasi grafis yang dihadirkan, maka akan membutuhkan sumber daya yang tinggi dan membutuhkan waktu tunggu untuk menampilkan fitur tersebut. Oleh karena itu dalam pengembangan aplikasi perlu mempertimbangkan aspek spesifikasi minimum yang dibutuhkan oleh aplikasi agar dapat bekerja dengan cepat.

Kedua, prinsip responsif mengacu kepada kemampuan tampilan antarmuka aplikasi untuk beradaptasi dengan cepat ketika tampil pada layar dengan ukuran yang beragam. Pada pengembangan *multiplatform* pengembang harus memahami konsep bahwa masing-masing perangkat yang digunakan oleh pengguna memiliki spesifikasi ukuran *display* yang berbeda, oleh karena itu aplikasi multimedia harus dibekali fitur untuk menyesuaikan tampilan antarmuka berdasarkan ukuran (resolusi) layar.

3.2 Elemen Antar Muka

Elemen antarmuka merupakan bagian-bagian yang menyusun sebuah halaman aplikasi, di dalamnya terdapat beberapa elemen yang saling melengkapi satu sama lain. Elemen antarmuka dalam buku ini dibedakan menjadi menjadi 4 (empat) kategori, yaitu :

1. *Input Control*

Elemen yang pertama adalah terkait dengan masukan yang diberikan oleh pengguna (*input*). Elemen tersebut memungkinkan pengguna untuk memasukkan informasi ke dalam sistem, seperti informasi tentang nama, sistem *login*, jawaban kuis dan sebagainya. *Input* pada umumnya ditampilkan dalam format *input text* atau dialog yang menunggu pengguna untuk memberikan perintah kepada aplikasi.



Gambar 14. input control multimedia interaktif “Inside Our Body” (Mustaqim, 2021)

Pada multimedia interaktif “Inside Our Body” karya Arzul Mustaqim (2021) memiliki sistem *input* yang mengharuskan pengguna untuk memasukkan namanya sebelum melanjutkan ke halaman berikutnya. Nama akan disimpan oleh aplikasi dan akan digunakan sebagai acuan dalam penilaian pada sistem evaluasi yang akan ditampilkan berikutnya.



Gambar 15. aplikasi multimedia interaktif “Cerita Afan” (Wandayanti, 2022)

Pada aplikasi multimedia interaktif “Cerita Afan” karya Mila Wandayanti (2022) memiliki sebuah fitur permainan teka-teki silang yang menggunakan *input text* untuk memasukkan jawaban atas pertanyaan yang ditampilkan oleh aplikasi.

Input control lainnya dapat diwujudkan dalam bentuk *check box*, *radio button*, *virtual keyboard*, *virtual joystick* dan beberapa format masukan lainnya.



Gambar 16. virtual joystick pada aplikasi Astro Farmer (Wibawanto, 2014)

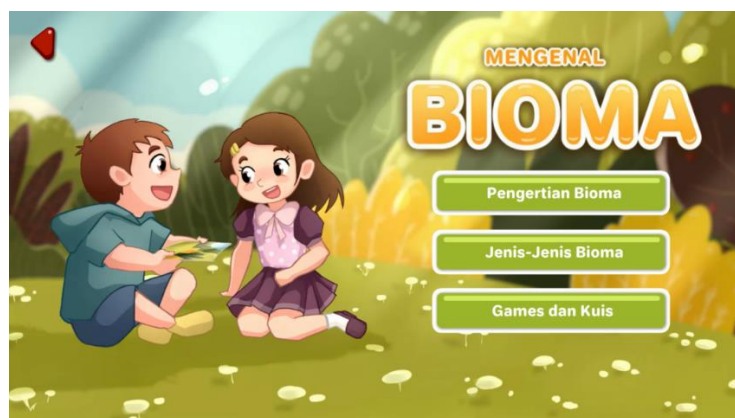
2. Navigation Component

navigation component atau komponen navigasi merupakan sebuah elemen antarmuka yang menjadi sarana interaksi pengguna bergerak di sebuah halaman, mengeksplorasi halaman atau berpindah halaman. Komponen navigasi yang paling umum adalah sebuah tombol. Tombol adalah elemen interaktif yang memungkinkan pengguna untuk mendapatkan umpan balik interaktif yang diharapkan dari sistem ketika menjalankan perintah tertentu. Pada dasarnya, tombol adalah elemen kontrol yang memungkinkan pengguna berkomunikasi langsung dengan aplikasi dan mengirimkan perintah yang diperlukan untuk mencapai tujuan tertentu. Misalnya, berpindah halaman, mengaktifkan fitur tertentu, mengunduh konten, memainkan video, dan banyak tindakan lain yang mungkin dilakukan. Salah satu alasan penggunaan tombol di setiap aplikasi digital termasuk multimedia interaktif adalah karena tombol secara efisien mampu meniru interaksi dengan objek di dunia fisik.

Sebuah tombol biasanya ditandai dengan jelas untuk visibilitas dan memiliki bentuk geometris tertentu dan seringkali didukung dengan label yang menjelaskan tindakan apa yang akan dilakukan melalui tombol tersebut. Secara umum dalam buku ini tombol dibedakan menjadi 5 jenis, yaitu :

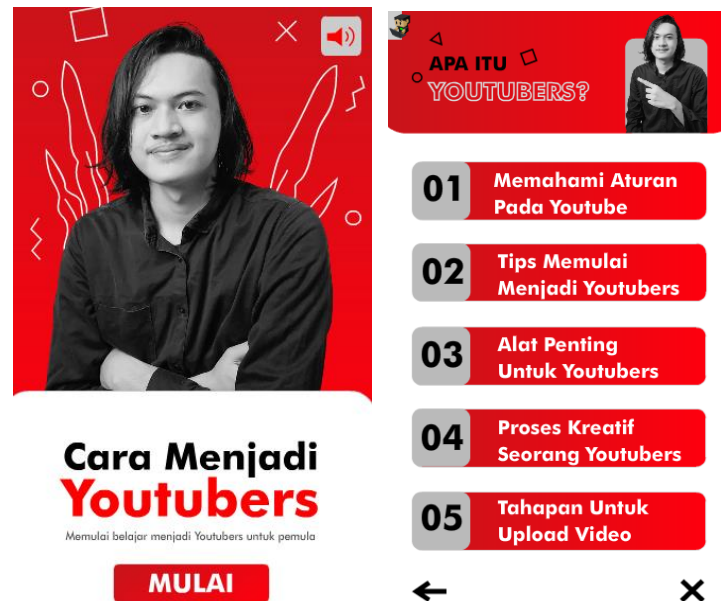
a. Tombol tindakan (*Call to Action Button / CTA button*)

tombol tindakan adalah elemen interaktif antarmuka pengguna yang ditujukan untuk mendorong pengguna melakukan tindakan tertentu. Tindakan ini menghadirkan konversi untuk halaman atau layar tertentu (misalnya mulai belajar, membaca materi, berlangganan, bermain dan sebagainya). Dengan kata lain, tujuan dari tombol tindakan adalah mengubah pengguna pasif menjadi aktif. Secara teknis semua tombol yang berada di sebuah halaman masuk ke dalam kategori tombol tindakan. Namun yang membedakannya dari semua tombol lain di halaman atau layar adalah sifatnya yang menarik, tombol harus didesain agar menarik perhatian dan merangsang pengguna untuk melakukan tindakan yang diperlukan.



Gambar 17. tombol tindakan pada multimedia interaktif “Bioma” (Amanata, 2022)

Pada multimedia interaktif “Bioma” karya Khonsa Amanata (2022) di atas, terdapat 3 buah tombol tindakan yang ditampilkan dengan bentuk dan warna lembut serta label tombol sesuai dengan masing-masing tindakan yang dapat dilakukan oleh pengguna.



Gambar 18. multimedia interaktif “Cara Menjadi Youtubers” (Halawah, 2022)

Pada multimedia karya Hadid Halawah (2022) yang membahas tentang cara menjadi Youtuber, 5 materi penting menjadi sebuah elemen tombol yang didesain secara *standout* dengan kontras warna yang tajam, merupakan sebuah penerapan prinsip desain yang tepat pada desain antarmuka.

b. Tombol teks (*text button*)

tombol teks adalah sebuah tombol yang dipresentasikan hanya dalam bentuk teks tanpa elemen visual pendukung lainnya. Teks tersebut harus menjelaskan tindakan yang akan terjadi jika pengguna mengklik atau mengetuk tombol. Tombol teks memiliki tingkat penekanan yang rendah dan biasanya digunakan untuk tindakan yang kurang penting dan tidak terlalu mengalihkan perhatian dari konten di sekitar.

Tombol teks sering kali ditemukan pada paragraf dengan kata-kata yang membutuhkan penjelasan-penjelasan khusus. Kata-kata khusus tersebut memiliki warna yang berbeda dan memiliki *hyperlink* yang dapat membawa pengguna menuju ke halaman yang menjelaskan secara detail terkait dengan kata-kata tersebut.

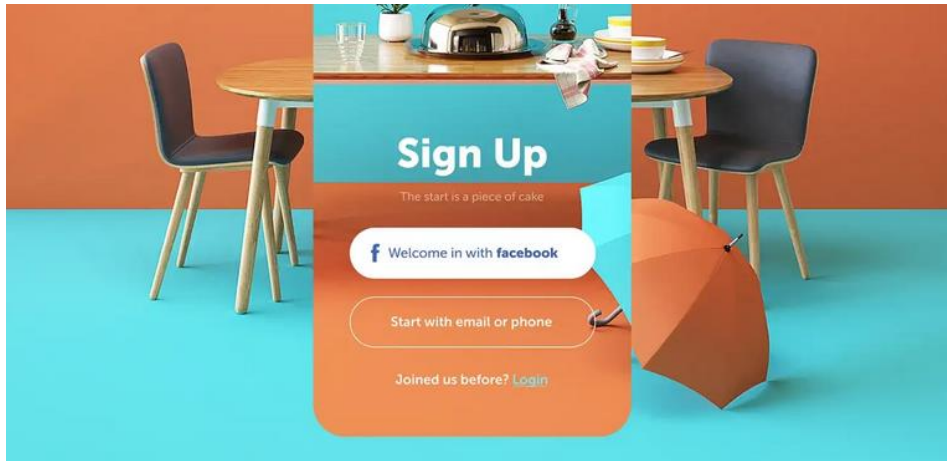


Gambar 19. tombol teks pada multimedia interaktif “Tur Planet” (Santoso, 2021)

Multimedia interaktif “Tur Planet” karya Baruna Kharisma Santoso (2021) memiliki beberapa halaman penjelasan tentang planet yang berada di sistem tata surya. Pada penjelasan masing-masing planet, terdapat tombol teks “SELENGKAPNYA”, yang apabila ditekan, maka pengguna akan mendapatkan penjelasan yang lebih detail berikut dengan video penjelasan. Tingkat urgensi tombol tersebut berada di hirarki yang tidak tinggi, dalam artian meskipun pengguna tidak menekan tombol tersebut, pengguna tetap akan dapat melihat informasi singkat dari sebuah planet yang dipilih.

c. Tombol Transparan (*Ghost Button*)

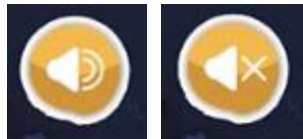
tombol jenis ini hanya memiliki label dan garis tepi (*outline*). Penggunaan tombol transparan merupakan sebuah tindakan peningkatan kompleksitas dan penekanan dari tombol teks dalam desain tombol. Tombol jenis ini biasanya menunjukkan tindakan yang penting tetapi bukan tindakan utama pada halaman.



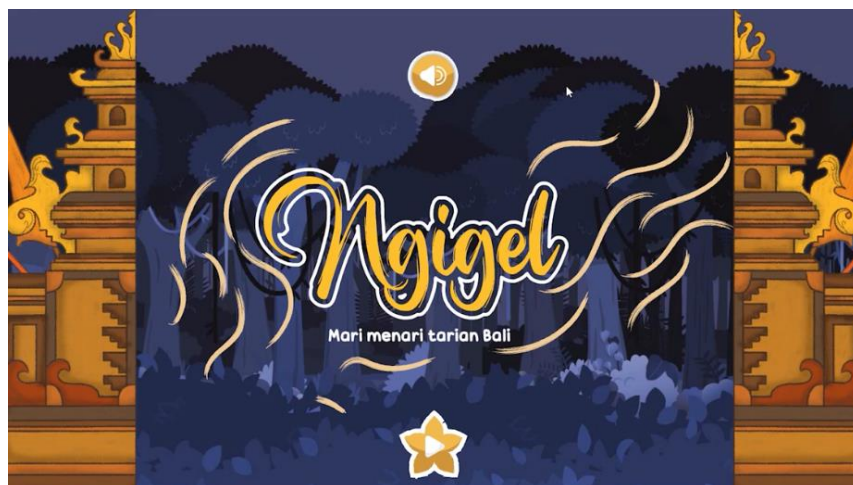
Gambar 20. tombol transparan “Start with email”

d. Tombol Saklar (*Toggle Button*)

tombol saklar digunakan untuk memilih salah satu dari dua kondisi yang ada. Pada tombol jenis ini, pilihan akan berubah menjadi lawan dari kondisi aktif ketika tombol ditekan. Sebagai contoh adalah tombol *On-Off* suara pada multimedia interaktif “Ngigel” karya Sabrina Putri (2021), satu buah tombol dapat berubah pada mode suara *on* dan suara *off*.



Gambar 21. tombol *onOff* suara



Gambar 22. peletakan tombol *toggle on off* suara pada multimedia interaktif “Ngigel” (Putri, 2021)

Beberapa literatur juga menambahkan jenis-jenis tombol yang lainnya seperti tombol aksi mengambang (*Floating Action Button*), tombol timbul (*raised button*), tombol ekspansi (*expandable button*) dan jenis lainnya.

3. *Information Component*

komponen informasi atau information component menampilkan informasi dalam bentuk tulisan, gambar, audio dan video. Dalam antarmuka aplikasi multimedia interaktif komponen informasi dapat berupa *tooltips* (label yang muncul pada saat *mouse* melintas di atas tombol, atau saat tombol ditekan), *progress bar* (tampilan untuk menunjukkan capaian pengguna akan aktivitas yang bertahap), notifikasi, *popup*, *loading*, dan sebagainya.

Sebagai contoh dalam multimedia interaktif “Rempah Indonesia” karya Rahmadila Karimatun Nisa (2021) pada saat tampilan peta Indonesia dihover dengan *mouse* , maka akan muncul sebuah label *popup* yang menunjukkan rempah-rempah yang terdapat pada daerah yang terseleksi, yang selanjutnya dapat ditekan dan akan menunjukkan detail informasi dari rempah terpilih.





Gambar 23. Komponen informasi pada aplikasi multimedia “Rempah Indonesia”
(Nisa, 2021)

4. *Container*

container adalah komponen antarmuka yang dapat menyatukan kumpulan lengkap komponen dan konten yang terkait satu sama lain sehingga dapat ditampilkan dan dapat digunakan secara keseluruhan. Pada elemen *container* elemen-elemen seperti teks, tombol, ilustrasi ditampilkan dalam sebuah kesatuan komposisi. Terdapat beberapa tipe *container*, yaitu :

a. Kartu informasi (*Card*)

panel kartu pada umumnya berbentuk persegi panjang atau modul persegi yang berisi informasi tertentu seperti tombol, teks, gambar, dan lainnya, sehingga menjadi satu kesatuan informasi. Panel tersebut ditampilkan secara konsisten baik dari segi warna, ukuran, penggunaan huruf dan peletakan tombol.

Sebagai contoh panel kartu yang ditampilkan pada multimedia interaktif “Vegefruit” karya Naelur Arzak (2021) yang menampilkan elemen teks penjelas, ilustrasi, tombol suara, dan tombol navigasi untuk melihat konten sebelum dan sesudahnya. Untuk lebih menarik perhatian, latar atau *background* sengaja dibuat sedikit transparan agar ilustrasi latar terlihat jelas. Di bagian depan juga ditambahkan ilustrasi karakter dengan gerakan dinamis sederhana seperti berkedip dan menggerakkan tangan.



Gambar 24. panel kartu informasi pada multimedia interaktif “Vegefruit” (Arzak, 2021)

b. Panel gulir (*Scroll Panel*)

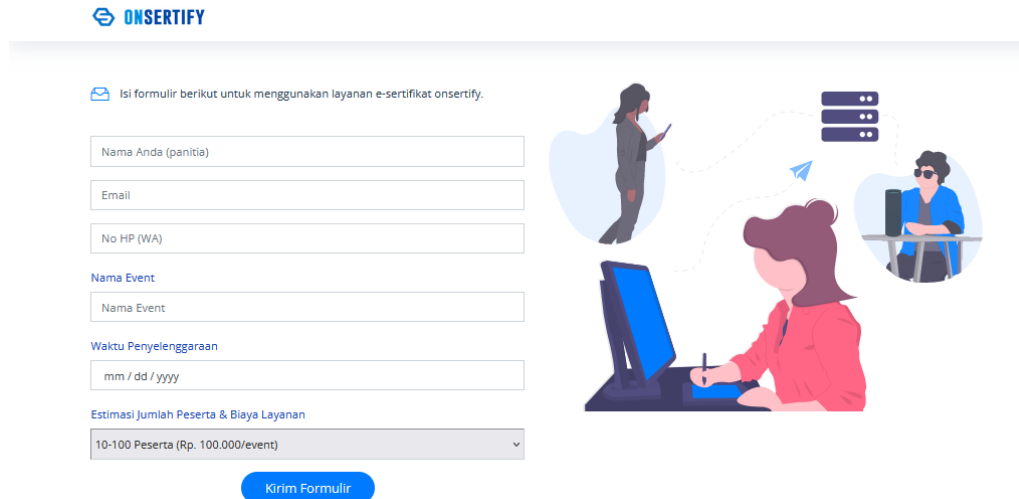
panel gulir adalah elemen antarmuka yang digunakan untuk membuat konten yang dapat digulir (*discroll*), hal tersebut memungkinkan pengguna untuk melihat lebih banyak konten gambar, kartu, atau konten lainnya dalam ruang halaman yang terbatas. Sebagai contoh, multimedia berjudul “Mengetahui Lapisan Bumi” karya Siti F.B. Oktafiardiani (2022) memiliki salah satu halaman yang menjelaskan tentang bagian lapisan bumi dengan teknik gabungan panel kartu informasi dan panel gulir. Teks penjelasan yang relatif banyak ditampilkan dalam bentuk *scroll* sehingga dapat menampilkan informasi yang lengkap.



Gambar 25. panel *scroll* pada multimedia “Mengetahui Lapisan Bumi” (Oktafiardiani, 2022)

c. Formulir (*form*)

formulir digunakan untuk mengambil data-data pengguna atau mendapatkan *input* tertentu dari pengguna. Sebagai contoh pada layanan e-sertifikat “onsertify” karya Wandah Wibawanto dan Pratama Bayu Widagdo (2020), ditampilkan beberapa data dalam bentuk formulir.

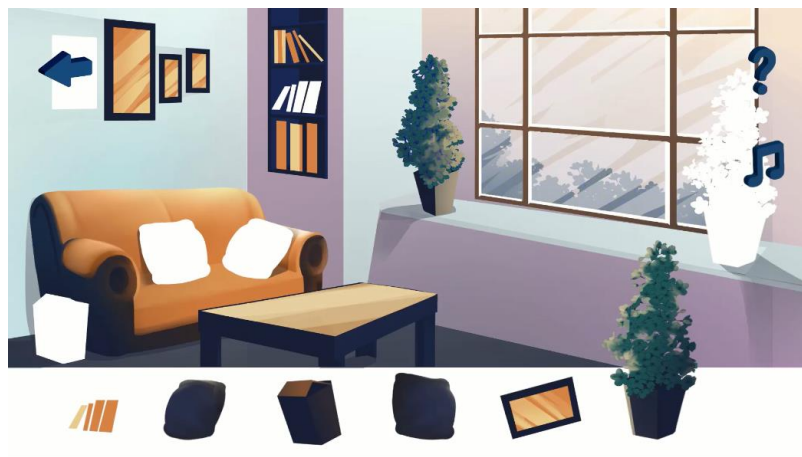


The image shows a web form for 'onsertify'. At the top left is the 'ONSERTIFY' logo. Below it is a blue envelope icon and the text 'Isi formulir berikut untuk menggunakan layanan e-sertifikat onsertify.' The form contains several input fields: 'Nama Anda (panitia)', 'Email', 'No HP (WA)', 'Nama Event', and 'Waktu Penyelenggaraan' (with a date format 'mm / dd / yyyy'). Below these is a dropdown menu for 'Estimasi Jumlah Peserta & Biaya Layanan' with the selected option '10-100 Peserta (Rp. 100.000/event)'. At the bottom center is a blue button labeled 'Kirim Formulir'. To the right of the form is an illustration of a woman in a red shirt sitting at a desk with a laptop, with other people and server icons in the background.

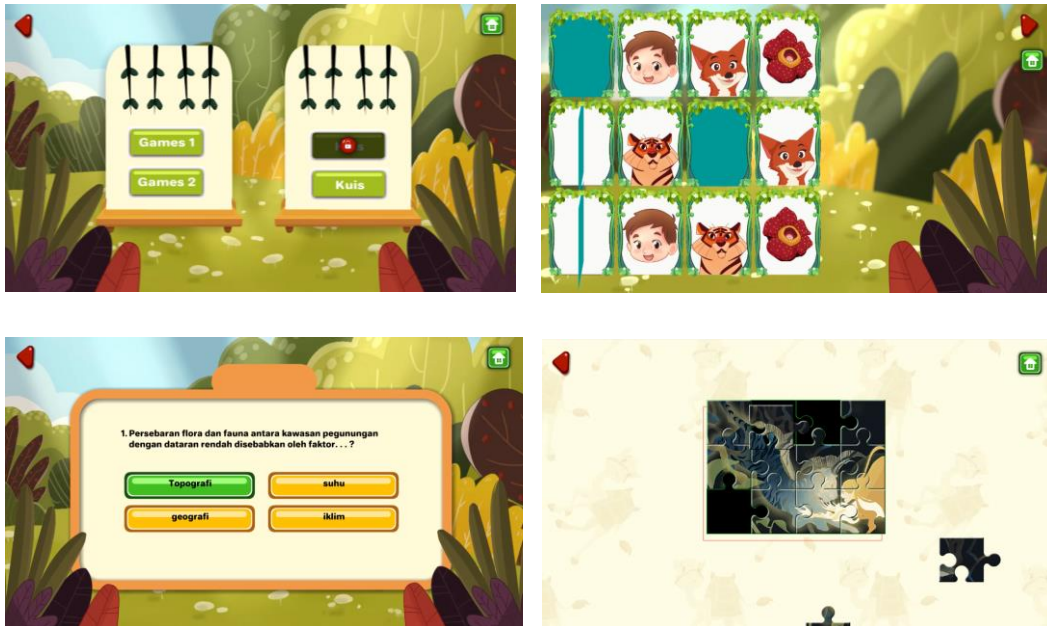
Gambar 26. formulir pada “onsertify” (Wibawanto dan Widagdo, 2020)

d. Permainan (*Game*)

Sebuah permainan dapat ditambahkan ke dalam aplikasi multimedia interaktif untuk menambah daya tarik. Sebuah permainan juga dapat dijadikan satu kesatuan objek dan dimasukkan ke dalam kategori *container*. Permainan yang dapat ditambahkan beragam mulai dari kuis, *puzzle*, permainan mencocokkan gambar, serta permainan yang bersifat edukatif pada aplikasi multimedia interaktif. Beberapa contoh berikut adalah permainan yang ditambahkan ke dalam aplikasi multimedia interaktif.



Gambar 27. permainan *drag and drop* pada multimedia “Cerita Afan” (Wandayanti, 2022)



Gambar 28. satu set permainan dalam multimedia “Bioma” (Amanata, 2022)

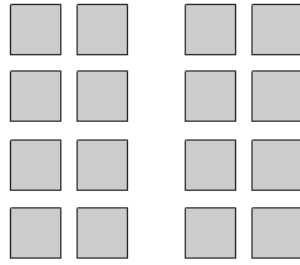
3.3 Tata Letak (*Layout*)

Desain tata letak (*layout*) adalah proses mengatur elemen visual—seperti teks, gambar, dan bentuk visual lainnya—pada suatu halaman tertentu. Desain tata letak sangat penting bagi proyek apa pun yang menyampaikan pesan melalui visual yang menarik, seperti tata letak majalah, desain situs *web*, iklan termasuk di dalamnya multimedia interaktif. Desain tata letak yang baik bersifat dinamis dan jelas, menciptakan poin visual yang menarik yang memandu pembaca melalui konten yang bertingkat dan diatur melalui prinsip-prinsip desain.

Beberapa desainer menerapkan teori *gestalt* dalam menyusun sebuah *layout*. Secara singkat teori *gestalt* didasarkan pada gagasan yaitu, ketika seseorang melihat suatu objek kompleks yang terdiri dari banyak elemen, mereka menerapkan metode sadar atau bawah sadar untuk mengatur bagian-bagiannya menjadi sistem yang terorganisir secara keseluruhan, bukan hanya sekumpulan objek sederhana. Oleh karena itu menata beberapa elemen tidak dapat dilakukan secara asal dan harus memperhatikan beberapa prinsip *gestalt*.

1. Kedekatan (*Proximity*)

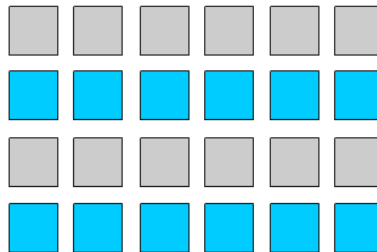
unsur-unsur yang dekat satu sama lain dianggap lebih terkait daripada unsur-unsur yang letaknya berjauhan. Dengan meletakkan beberapa elemen yang saling berdekatan, maka akan dianggap satu kelompok tertentu. Sebagai contoh di bawah ini, dengan mengatur jarak elemen persepsi yang dilihat oleh pengguna adalah terdapat 2 kelompok atau 2 kolom.



Gambar 29. prinsip *proximity*

2. Kesamaan (*Similarity*)

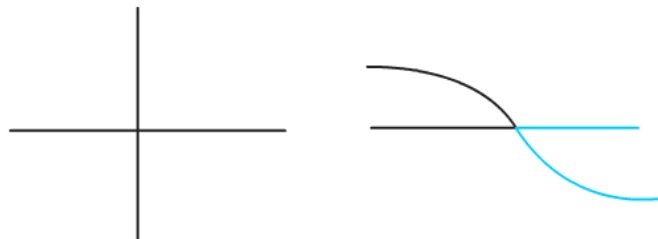
seseorang akan mengasosiasikan dan memperlakukan beberapa elemen grup yang memiliki karakteristik visual yang konsisten sebagai sebuah kelompok. Karakteristik visual dapat dibentuk melalui bentuk yang sama atau warna yang sama. Sebagai contoh di bawah ini, dengan mengelompokkan warna, maka akan terbentuk persepsi 2 kelompok secara horisontal.



Gambar 30. prinsip *similarity*

3. Kontinuitas (*Continuity*)

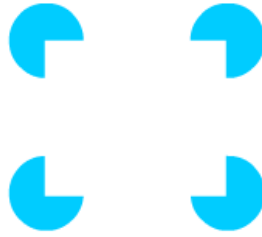
secara sadar atau bawah sadar, seseorang akan lebih menyukai dan mempresepsikan garis atau jalur yang berkesinambungan dan tidak terputus ketika melihat suatu pola tertentu. Pada 2 contoh berikut, persepsi yang didapatkan adalah adanya 2 garis atau 2 kurva yang saling berpotongan, bukan 4 buah garis atau kurva yang bertemu pada satu titik.



Gambar 31. prinsip *continuity*

4. Penutupan (*Closure*)

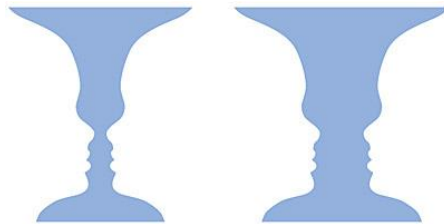
persepsi manusia memiliki kecenderungan yang kuat untuk melihat figur yang lengkap, bahkan ketika kontur figur tersebut rusak atau ambigu. Sebagai contoh, mata manusia dapat melihat persegi panjang putih di atas empat lingkaran, dan bukan empat lingkaran yang masing-masing memiliki bagian yang hilang.



Gambar 32. prinsip *closure*

5. Hubungan *figure-ground*

dalam prinsip ini persepsi seseorang dapat bergantian antara dua kemungkinan interpretasi dari bidang visual yang sama. Contoh yang paling umum adalah gambar Rubin Vas di bawah ini, yaitu gambar yang dapat dipresepsikan sebagai vas atau dua wajah, tetapi tidak dapat melihat keduanya sekaligus. Gambar wajah disebut sebagai *figure* dan akan terlihat jelas ketika jarak antara keduanya agak dekat, sementara gambar vas disebut sebagai *ground* akan terlihat jelas ketika jarak gambar wajah menjauh.



Gambar 33. Rubin Vas yang menunjukkan hubungan *figure-ground*

6. Keterhubungan yang seragam (*Uniform connectedness*)

keterhubungan seragam mengacu pada hubungan elemen yang didefinisikan dengan melingkupi elemen di dalam elemen lain, wilayah, atau pengaturan area dari suatu halaman.



Gambar 34. keterhubungan

Pada contoh di atas (kiri), keterhubungan membuat kita dapat mengelompokkan objek berdasarkan warna, meskipun memiliki bentuk dan ukuran yang sama. Pada contoh gambar (tengah), keterhubungan memungkinkan kita membuat atau mengorganisasikan konten dalam keteraturan pengelompokan. Pada contoh gambar (kanan) penggabungan beberapa prinsip yaitu kesamaan dan penutupan membentuk sebuah kelompok yang mudah untuk dikenali.

3.3.1 Aspect Ratio

Aspect ratio (aspek rasio) adalah hubungan proporsional antara lebar dan tinggi gambar yang secara umum ditulis sebagai rumus perbandingan lebar ke tinggi. Dalam pengembangan aplikasi yang melibatkan grafis secara intens seperti multimedia interaktif, aspek rasio menjadi pertimbangan penting sebelum memulai perancangan aplikasi.

Resolusi layar *platform* yang akan dituju oleh aplikasi pada saat ini sangat beragam, dan sangat sulit seorang pengembang aplikasi membuat penyesuaian setiap saat. Pada prinsip *UI* responsif pada bab sebelumnya dijelaskan bahwa aplikasi yang baik mampu beradaptasi dengan ukuran layar berbagai *platform*. Teknik yang biasa digunakan oleh pengembang *website* adalah membaca terlebih dahulu ukuran layar, kemudian mengatur letak masing-masing elemen antarmuka berdasarkan ukuran layar tersebut. Akan tetapi dalam pengembangan multimedia interaktif yang lebih kompleks, teknik tersebut relatif lebih rumit untuk dipraktikkan, sehingga diperlukan metode yang lebih sederhana namun efektif.

Dalam memahami aspek rasio dan resolusi layar yang akan digunakan, terlebih dahulu dikonsepkan secara detail terkait posisi layar yang akan digunakan. Semenjak *platform mobile* (Android, IOS atau yang lain) populer, maka posisi layar tidak selalu *landscape* (mendatar) sebagaimana monitor komputer atau layar televisi. Aplikasi dengan posisi *portrait* (tegak) juga sering kali ditemui. Sementara aplikasi multimedia interaktif yang dapat bekerja pada dua mode (*landscape* dan *portrait*) tidak terlalu umum, pengembang pada umumnya hanya memilih satu posisi saja.



Gambar 35. multimedia interaktif “Panduan Umrah” dengan posisi *portrait* (Zuhri, 2020)



Gambar 36. multimedia interaktif “Bendera Dunia” dengan posisi *landscape* (Sivana, 2022)

Aspek rasio yang secara umum dapat ditemui antara lain adalah 3:2, 4:3, 1:1, 16:9 dan 2:1. Sebagai contoh, apabila kita akan mengembangkan aplikasi yang akan bekerja pada *platform mobile* (pada saat buku ini ditulis, sebagian besar aspek rasio *mobile phone* adalah 2:1), maka kita dapat menentukan resolusi layar dalam satuan *pixel* yaitu 1440 *pixel* x 720 *pixel* atau kelipatan di atasnya. Resolusi dalam satuan *pixel* selanjutnya akan dilakukan penskalaan (*scaling*) secara otomatis oleh sistem, baik itu diperbesar atau diperkecil.

Dalam beberapa literatur, untuk mendapatkan aspek rasio yang mendekati *golden ratio* (1, 618) digunakan perbandingan 16:9. *Golden ratio* adalah sebuah perbandingan yang dirasa memiliki kadar estetik tertinggi. Ketika sebuah desain menggunakan perbandingan *golden ratio*, maka desain dinilai lebih mudah mencapai keindahan. Meskipun angka *golden ratio* adalah angka yang tidak pasti, namun demikian mendesain tampilan antarmuka dengan pendekatan *golden ratio* akan mempermudah pengembang dalam memperoleh tampilan yang menarik. Berikut perbandingan rasio 2:1, 16:9 dan *golden ratio*.



Gambar 37. aspek rasio

3.3.2 Sistem *Grid*

Sistem *grid* adalah alat bantu yang digunakan desainer untuk membangun desain, mengatur informasi, dan membuat pengalaman pengguna yang konsisten. *Grid* dibentuk oleh beberapa garis bantu untuk membantu konsistensi peletakan elemen antarmuka. Sistem *grid* akan membantu dalam menyelaraskan elemen layar berdasarkan kolom dan baris berurutan. Garis-garis bantu dalam format baris dan kolom akan digunakan untuk menyusun teks, gambar, dan elemen interaktif lainnya secara konsisten di seluruh halaman. Teknik dasar sistem *grid* antara lain:

1. *Rule of Third*

sistem *grid* ini membagi konten secara merata menjadi tiga bagian, secara horizontal dan vertikal. Elemen desain dapat diletakkan di persimpangan garis pemisah tersebut atau di sepanjang salah satu garis. *Rule of third* efektif untuk membangun keseimbangan simetris maupun asimetris. Sebagai contoh, dalam pengembangan multimedia interaktif “Batik Kita” karya Zunalia Akhmad (2020), pada halaman judul, layar dibagi menjadi 3 bagian sesuai dengan aturan *rule of third*. Elemen judul diletakkan pada sepertiga pertama di bagian atas, dan elemen navigasi (2 tombol belajar batik dan bermain batik) diletakkan di sepertiga pertama bagian bawah. Sebagai *empashis* (daya tarik) diletakkan ilustrasi pada 2 per 3 bagian berikutnya.



Gambar 38. *rule of third* pada halaman judul multimedia interaktif “Batik Kita” (Akhmad, 2020)

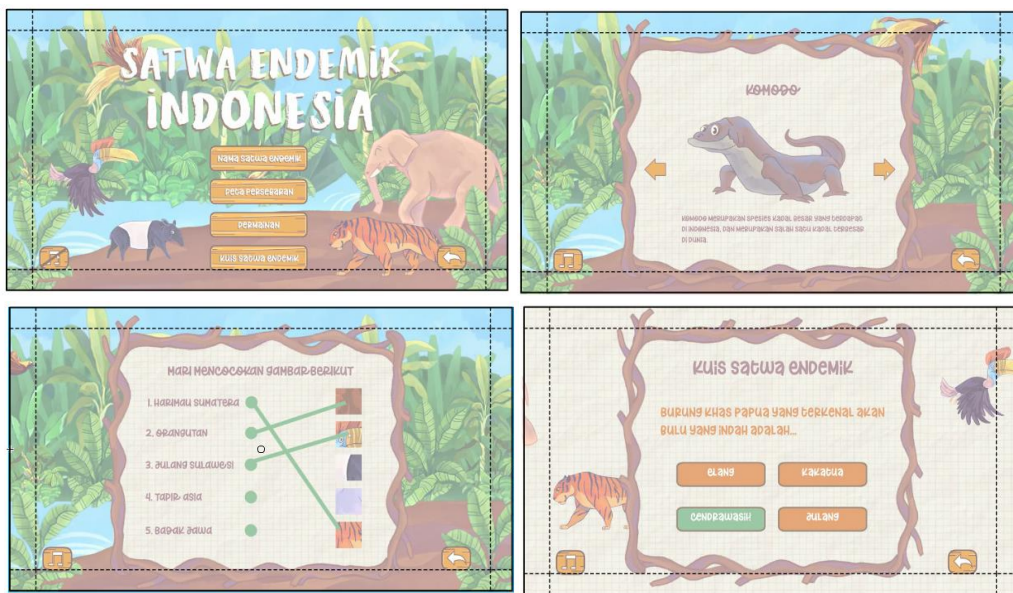
Setelah desain dirancang dengan pendekatan *rule of third*, maka finalisasi desain akan terlihat lebih dinamis, memiliki proporsi yang mudah diterima dan mudah dipahami oleh pengguna.



Gambar 39. finalisasi penerapan *rule of third* pada halaman judul multimedia interaktif “Batik Kita” (Akhmad, 2020)

2. Kolom Tunggal

grid kolom tunggal merupakan sistem *grid* yang paling sederhana yaitu terdiri dari satu kolom yang dikelilingi oleh sebuah jarak (*margin*). Penggunaan kolom tunggal akan membentuk keseimbangan simetris dan memusatkan perhatian pada bagian tengah halaman. Mengelola desain dengan sistem kolom tunggal tidak terlalu kompleks dan mudah untuk mendapatkan hasil yang standar. Sebagai contoh multimedia interaktif “Satwa Endemik Indonesia” karya Kamal Saad (2021), pada setiap halamannya menggunakan teknik *grid* kolom tunggal. Konten diletakkan di tengah layar secara konsisten dari halaman satu ke halaman lainnya. Pada bagian latar (*background*) ditambahkan ilustrasi yang sesuai dengan tema aplikasi sebagai elemen estetis.



Gambar 40. penggunaan *grid* kolom tunggal pada multimedia interaktif “Satwa Endemik Indonesia” (Saad, 2021)

3. Multi-Kolom

grid tipe multi-kolom menyediakan format yang fleksibel untuk tata letak dengan hierarki yang kompleks atau yang mengintegrasikan teks, ilustrasi, dan elemen lainnya. Semakin banyak kolom yang dibuat peletakan elemen menjadi lebih fleksibel. *Grid* dapat digunakan untuk mengartikulasikan hierarki elemen dengan mengelompokkan zona untuk konten-konten sejenis. Dalam praktik tata letak multi kolom, tidak semua ruang harus diisi. Penggunaan *whitespace* (ruang kosong) akan menghasilkan desain yang sifatnya ringan, minimalis dan lebih mudah mendapatkan fokus pada konten yang ada.

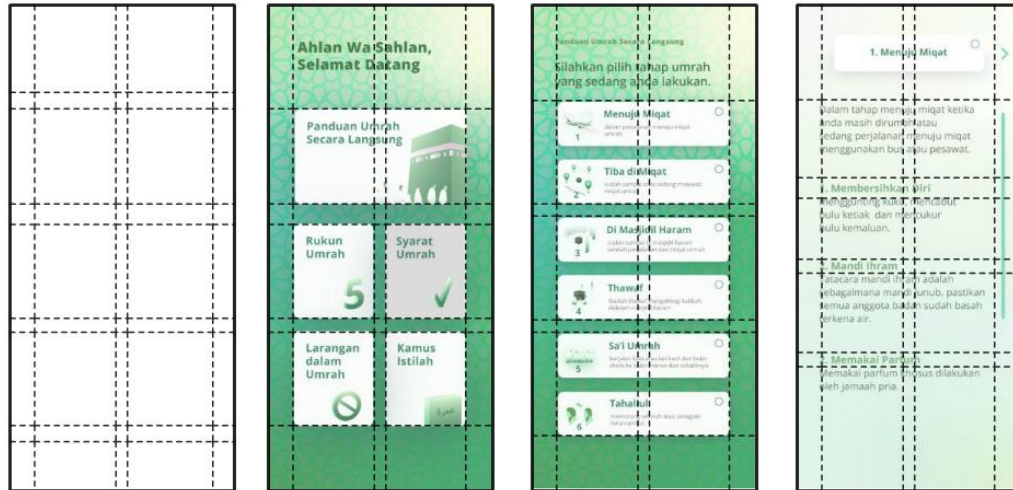
Sebagai contoh multimedia interaktif “Mengenal Rempah Indonesia” karya Rahmadila Karimatun Nisa (2021), menggunakan beberapa kombinasi grid multi-kolom untuk menampilkan konsistensi halaman yang dinamis.



Gambar 41. penggunaan *grid* multikolom pada multimedia interaktif “Mengenal Rempah Indonesia” (Nisa, 2021)

4. Modular

tipe *grid* modular memiliki pembagian horizontal yang konsisten dari atas ke bawah serta pembagian vertikal dari kiri ke kanan. Garis-garis bantu akan mengatur penempatan dan pemangkasan gambar dan teks. Peletakan secara modular hampir mirip dengan penempatan multi-kolom, hanya saja pada *grid* modular, ukuran kolom dan baris dibuat seragam. Elemen-elemen yang diletakkan pada masing-masing area harus mematuhi garis bantu untuk memperoleh komposisi yang baik.



Gambar 42. penggunaan grid modular pada multimedia interaktif “Panduan Umrah” (Zuhri, 2020)

3.3.3 Eyeflow

Eyeflow merupakan bagaimana mata diarahkan melalui sebuah desain, di mana mata pengguna melihat pertama kali, ke mana selanjutnya, di mana mata berhenti sejenak, dan berapa lama mata bertahan. Terdapat beberapa pola *eyeflow* yang dapat diambil dari berbagai referensi, diantaranya adalah :

1. Diagram Gutenberg

dalam pola ini pandangan mata umumnya melihat mulai dari kiri atas ke kanan bawah, sudut kanan atas adalah area dengan atensi terkuat, sedangkan sudut kiri bawah adalah area dengan atensi terendah. Pada contoh multimedia interaktif “Mengenal Tokoh Uang” karya Bintang Falikh (2020) pada halaman sebagian besar halamannya menerapkan *eyeflow* dengan pendekatan diagram Gutenberg. Arah mata diawali dari pojok kiri atas dengan ilustrasi mata uang diikuti dengan judul aplikasi dengan pada area atensi kuat (pojok kanan atas), dilanjutkan dengan peletakan tombol pada terminal area sebagai tempat pemberhentian mata. Pada bagian pojok kiri bawah hanya diletakkan ilustrasi pemanis sebagai kelanjutan dari fokus utama ilustrasi.

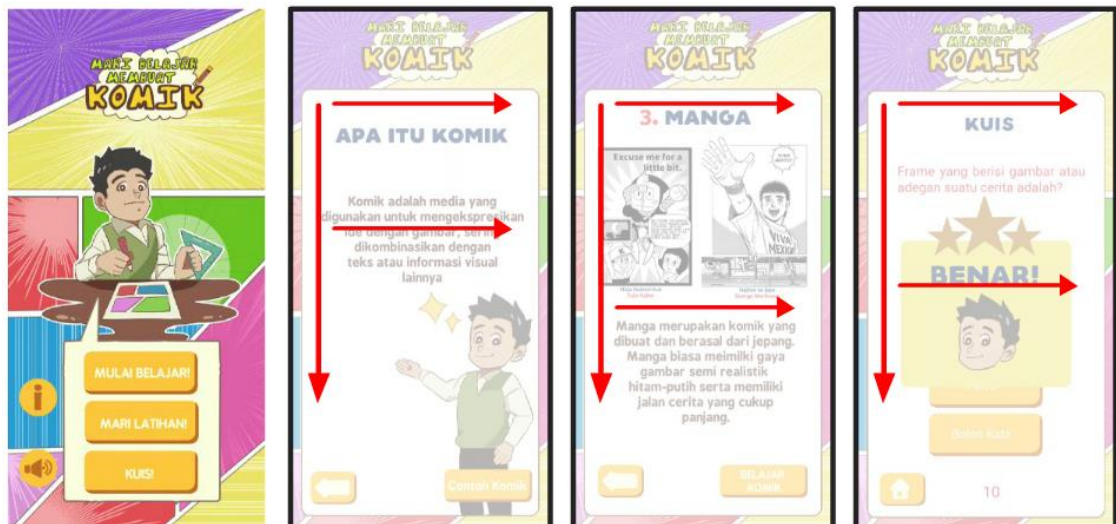




Gambar 43. *eyeflow* dengan pendekatan diagram Gutenberg pada multimedia interaktif “Mengenal Tokoh Uang (Falikh, 2020)

2. Tata letak pola-F

dalam pola ini pandangan mata dimulai di kiri atas dan bergerak melintasi halaman ke kanan sebelum bergerak sedikit ke bawah dan mengulangi gerakan melintasi halaman. Pola umumnya mengikuti bentuk huruf F. Ciri utama dari tata letak pola F adalah adanya repetisi elemen secara horisontal setidaknya 2 kali di bagian atas dan dilanjutkan dengan elemen penting (seperti tombol navigasi) di bagian bawah. Sebagai contoh, multimedia interaktif “Belajar Membuat Komik” karya Felife Satria (2020), dengan posisi aplikasi yang tegak (*portrait*), elemen pada sebagian besar halaman ditampilkan dengan pola 2 baris horisontal dan dilanjutkan dengan tombol navigasi di bagian bawah.

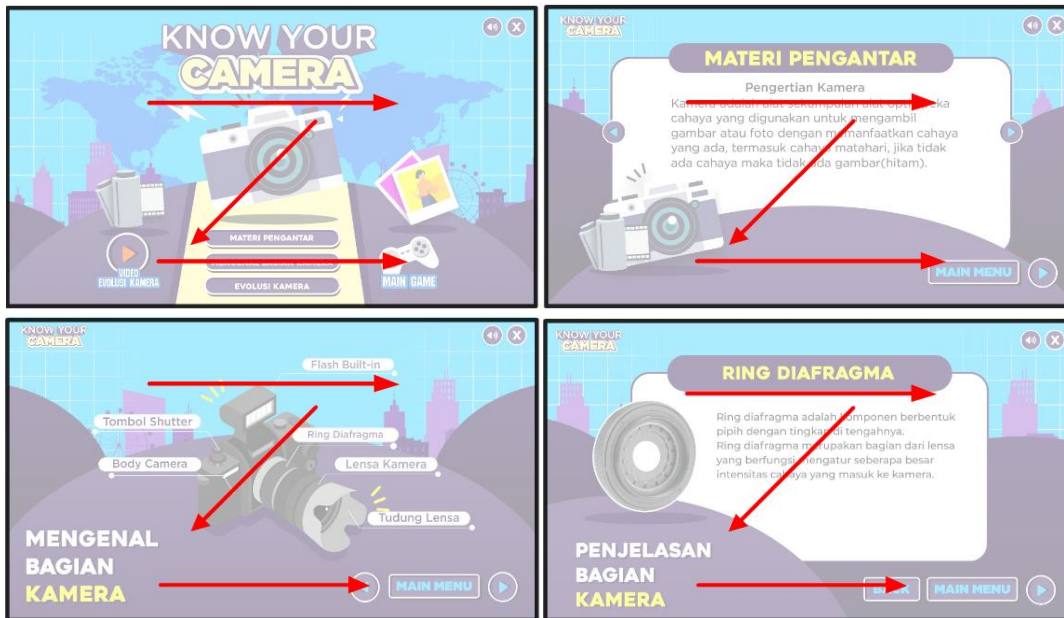


Gambar 44. *eyeflow* dengan pola-F pada multimedia interaktif “Belajar Membuat Komik” (Satria, 2020)

3. Tata letak pola Z

pada pola ini arah pandang mata dimulai dari kiri atas dan bergerak ke kanan. Pada pojok kanan atas bergerak ke bawah dan ke kiri secara diagonal sebelum bergerak sekali lagi ke

kanan. Secara keseluruhan itu mengikuti bentuk huruf Z dan mengulang pola di bawah halaman hingga mencapai kanan bawah. Sebagai contoh multimedia interaktif “Know Your Camera” karya Yusuf Akbar (2021) yang sebagian besar tata letak halamannya menggunakan *eyeflow* pola Z.



Gambar 45. *eyeflow* dengan pola-Z pada multimedia interaktif “Know Your Camera” (Akbar, 2021)

4. Tata letak pola O / C

pada tata letak ini elemen disusun menyerupai huruf O atau huruf C, sehingga arah pandang mata dibuat melingkar. Dalam multimedia interaktif, tata letak pola O/C pada umumnya dikombinasikan dengan pola lain, dan digunakan pada halaman khusus yang bertujuan untuk mendapatkan tampilan estetik yang menarik. Sebagai contoh pada multimedia interaktif “Mengenal P3K” karya Ananda Firdaus (2021) yang menampilkan alat-alat P3K dalam format melingkar. Demikian halnya dengan multimedia interaktif berjudul “4 Sehat 5 Sempurna” karya Winda Putri (2022) yang menampilkan elemen makanan dalam format melingkar.



Gambar 46. multimedia interaktif “Mengenal P3K” dan “4 Sehat 5 Sempurna” Menggunakan pendekatan pola O/C (Firdaus, 2021; Putri, 2022)

3.3.4 Fine Layout halaman multimedia interaktif

Setelah memahami prinsip-prinsip dasar dalam pengembangan aset visual, tahapan selanjutnya adalah mengubah *rough layout* (sketsa kasar) yang telah di buat pada tahapan sebelumnya menjadi *fine layout*. Pengembangan aset visual dapat dikerjakan dengan menggunakan aplikasi grafis seperti Photoshop, Corel Draw, Adobe Illustrator, Procreate, Adobe Animate atau aplikasi grafis apapun yang dikuasai oleh tim pengembang. Perhatikan contoh-contoh transformasi tampilan visual dari *rough layout* menjadi *fine layout* berikut :

1. Multimedia interaktif Batik Kita



Gambar 47. multimedia “Batik Kita” (Ahmad, 2020)

Pada desain multimedia interaktif di atas, terlihat terdapat beberapa perubahan yang terjadi dari *rough layout* menuju ke *fine layout*. Perubahan tersebut wajar adanya karena hanya bersifat *minor*. Pada halaman “Puzzle Batik”, terdapat perubahan yang cukup signifikan karena dirasa tampilan baru lebih ringkas dan lebih menarik untuk memulai sebuah permainan interaktif.

2. Multimedia “Bersih-Bersih”



Gambar 48. multimedia “Bersih-bersih” (Lidya, 2021)

Pada desain multimedia di atas, penyesuaian yang dilakukan dari *rough layout* menuju ke *fine layout* sangat sedikit, karena desain *rough layout* sudah dibuat dengan baik dan detail. Penambahan-penambahan detail seperti tekstur dan *paneling* menjadikan *fine layout* yang dihasilkan sangat baik.

3. Multimedia “Ayo Memilah Sampah”



Gambar 49. multimedia “Ayo Memilah Sampah” (Setianingrum, 2022)

Pada desain multimedia interaktif di atas, pengembang grafis sengaja tidak menambahkan teks pada aset visual yang dibuat. Dalam pengembangan sebuah aplikasi multimedia, teks bisa ditambahkan secara dinamis pada saat proses pengkodean. Sebagai contoh, untuk mendapatkan tombol yang sifatnya seragam, maka cukup dibuat latar dari tombol tersebut, sementara teks bisa ditambahkan pada tahapan selanjutnya.

4. Multimedia Fire Fight



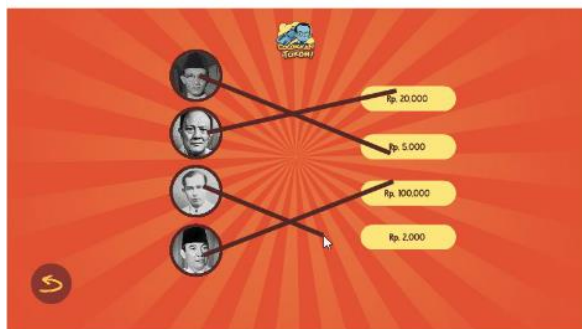
Gambar 50. multimedia “Fire Fight” (Fahlevi, 2022)

Pada multimedia interaktif di atas, sangat syarat dengan gambar ilustrasi. Kapasitas tim pengembang grafis harus mampu memvisualisasikan konsep dengan baik untuk mendapatkan tampilan visual yang tepat.

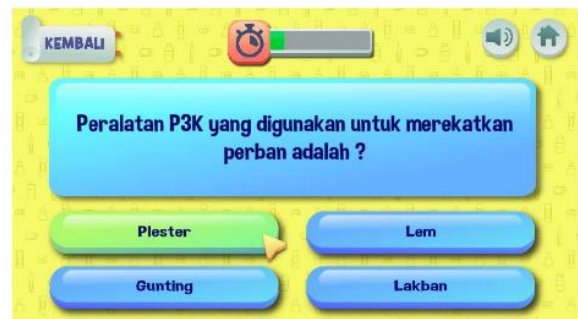
Selain contoh-contoh di atas, berikut ditampilkan contoh-contoh tata letak multimedia interaktif yang memiliki kualitas baik dan memenuhi prinsip-prinsip desain yang baik serta prinsip pengembangan antarmuka yang baik.



Gambar 51. multimedia interaktif “Kuliner Khas Nusantara” (Abdullah, 2021)



Gambar 52. multimedia interaktif “Mengenal Tokoh Uang” (Falikh, 2020)



Gambar 53. multimedia interaktif “Mengenal P3K” (Firdaus, 2021)



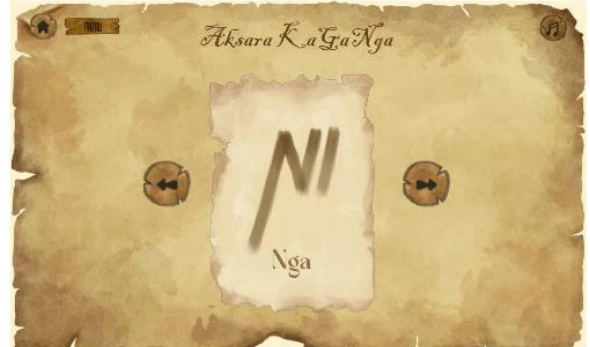
Gambar 54. multimedia interaktif “Know Your Camera” (Akbar, 2021)



Gambar 55. multimedia interaktif “Vegefruit” (Razak, 2021)



Gambar 56. multimedia interaktif “Bersih-bersih Yuk” (Lidya, 2021)



Gambar 57. multimedia interaktif “Menjelajah Bumi Raflesia” (Salsabila, 2021)



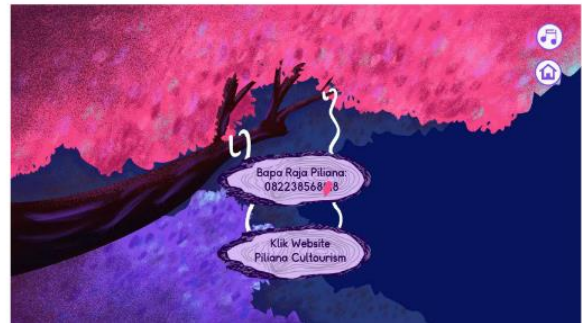
Gambar 58. multimedia interaktif “Rempah Indonesia” (Nisa, 2021)



Gambar 59. multimedia interaktif “Sinau Batik” (Amin, 2021)



Gambar 60. multimedia interaktif “Ayo Belajar Tata Surya” (Priyambodo, 2021)



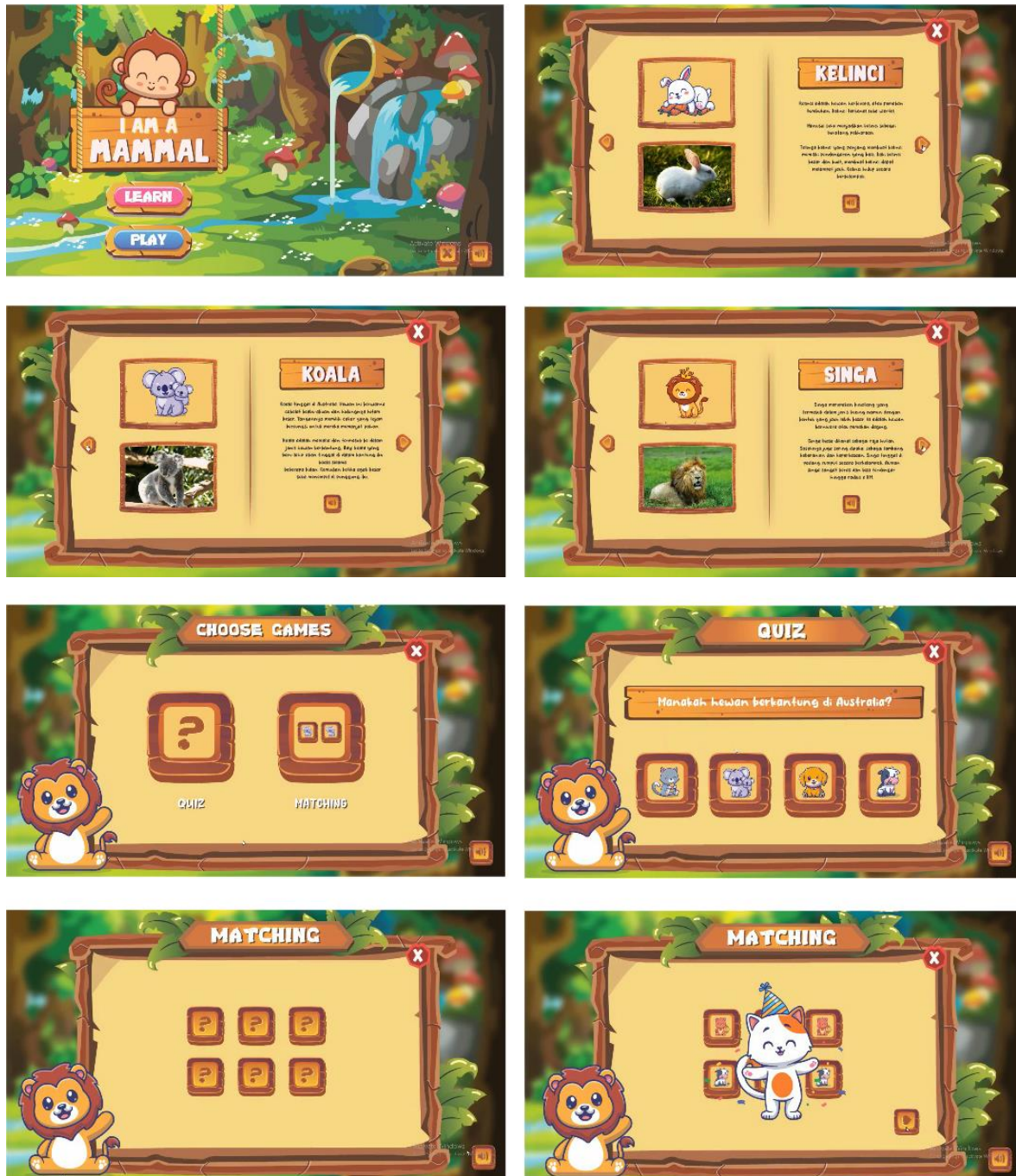
Gambar 61. multimedia interaktif “Piliana Cultourism” (Rosalind, 2021)



Gambar 62. multimedia interaktif “Ngigel” (Putri, 2021)



Gambar 63. multimedia interaktif “Lincih Matika” (Setyaningrum, 2021)



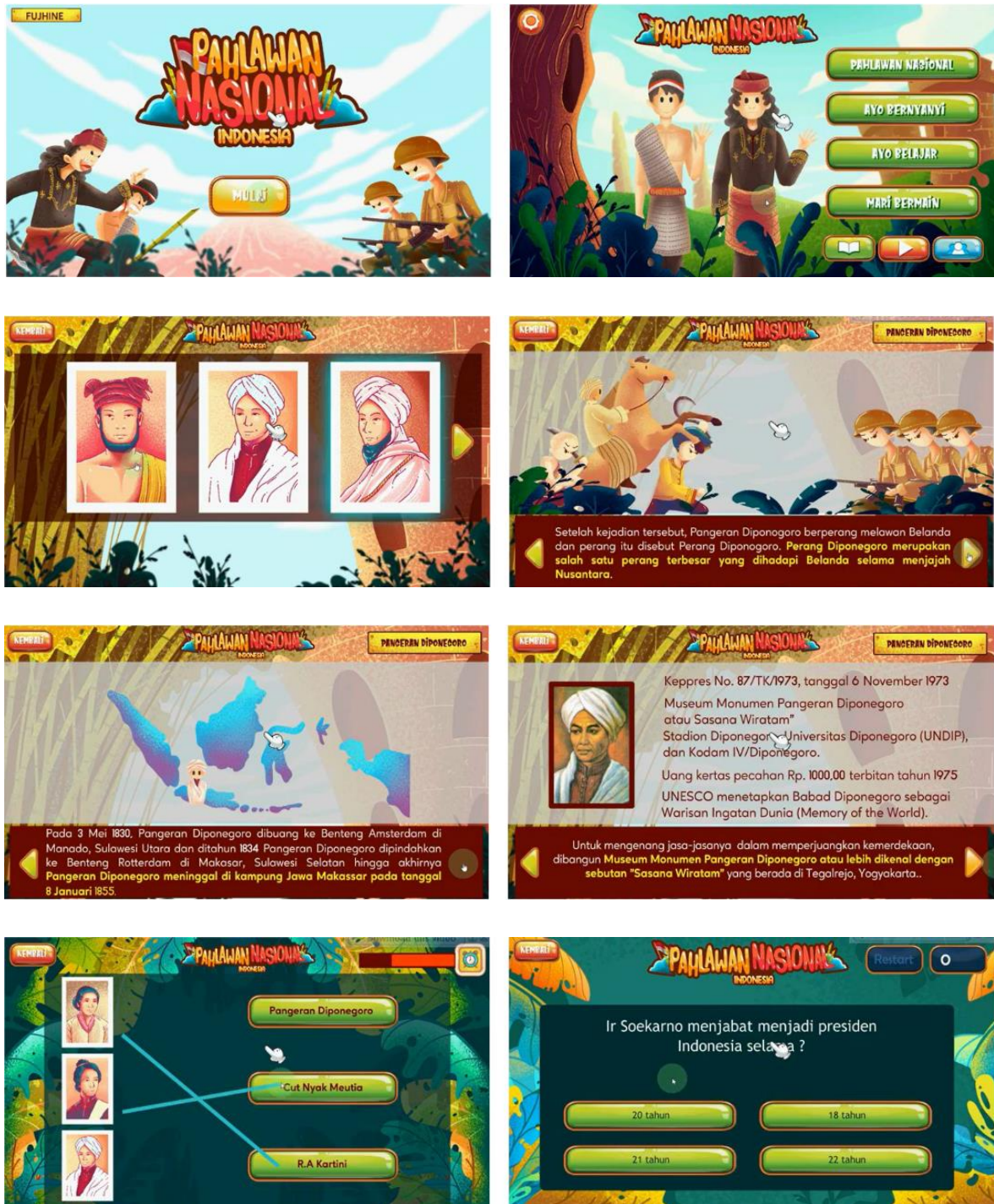
Gambar 64. multimedia interaktif “I am a mammal” (Firly, 2022)



Gambar 65. multimedia interaktif “Mengenal Bioma” (Amanata, 2022)



Gambar 66. multimedia interaktif "Scuba Diving" (Azahra, 2018)



Gambar 67. multimedia interaktif “Pahlawan Nasional Indonesia” (Novrizal, 2020)



Gambar 68. multimedia interaktif “Pakaian dan Rumah Adat Nusantara” (Murtiningsih, 2013)

BAB 4

Adobe Animate

Sebelum melanjutkan ke tahapan selanjutnya, terlebih dahulu perlu dikenal aplikasi atau perangkat lunak yang akan digunakan dalam pengembangan aplikasi multimedia di buku ini. Secara umum pengembangan aplikasi multimedia dapat menggunakan berbagai macam aplikasi, namun penulis sengaja menggunakan Adobe Animate sebagai kesinambungan antara buku-buku sebelumnya dan kesinambungan dengan materi tutorial yang ada di *website* maupun *channel* Youtube penulis.

Adobe Animate merupakan sebuah aplikasi *Authoring Tool* atau pengelola multimedia. Adobe Animate secara spesifik dapat digunakan untuk merancang grafis berbasis vektor, animasi, aplikasi *web*, dan aplikasi komputer maupun gawai. Animate memiliki dukungan terhadap grafik bertipe *Bitmap*/raster, pengelolaan teks, audio dan video, serta memiliki dukungan bahasa pemrograman *Actionscript* yang dapat digunakan untuk menunjang interaktivitas sebuah aplikasi.

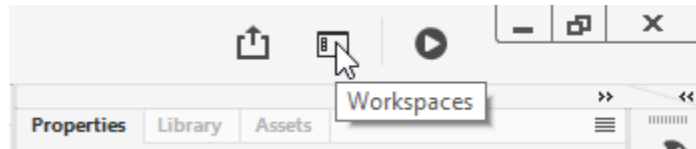
4.1 Area Kerja Adobe Animate

Adobe Animate memiliki area kerja yang cukup kompleks namun mudah untuk dipahami. Area kerja Adobe Animate secara *default* disebut sebagai mode *esensial*. Tampilan pada mode *esensial* dapat dilihat pada gambar berikut :



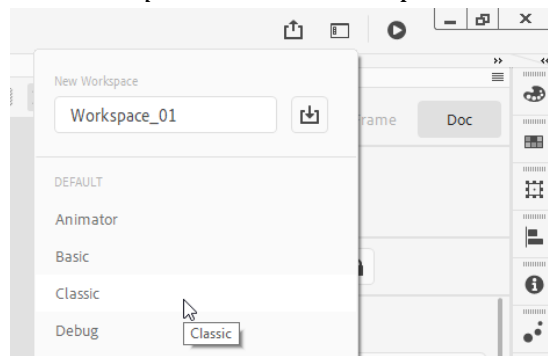
Gambar 69. area kerja Adobe Animate pada mode *esensial*

Adobe Animate, merupakan *brand* baru dari Adobe Flash dan Macromedia Flash. Meskipun memiliki tampilan yang relatif berbeda dengan Adobe Flash atau Macromedia Flash, pengguna yang terbiasa dengan aplikasi lama dapat mengatur ulang tampilan menjadi mode *classic* dengan menekan tombol *workspace* yang berada di bagian kanan atas.



Gambar 70. lokasi pengaturan area kerja (*workspaces*)

Pada buku ini akan dijelaskan dengan tampilan kerja mode *classic*, agar bagi pengguna Adobe Flash tetap dapat mengikuti tutorial-tutorial di buku ini. Alasan lain penggunaan mode *classic* adalah banyaknya tutorial-tutorial Adobe Flash yang sangat representatif dan tetap dapat digunakan sampai saat buku ini ditulis. Selanjutnya untuk mengubah tampilan menjadi mode *classic*, cukup menekan menu *workspaces* dan memilih opsi *classic*.



Gambar 71. opsi mode *classic*

Pada mode *classic* tampilan Adobe Animate sebagian besar sama dengan tampilan Adobe Flash atau Macromedia Flash, sebagai berikut :

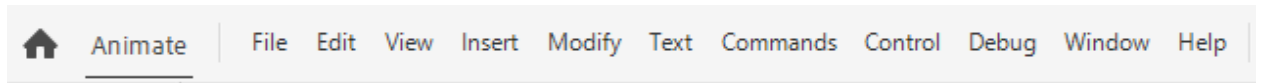


Gambar 72. tampilan pada mode *classic*

Area Kerja pada Adobe Animate dapat dijabarkan secara sederhana sebagai berikut :

4.1.1 Menu Utama

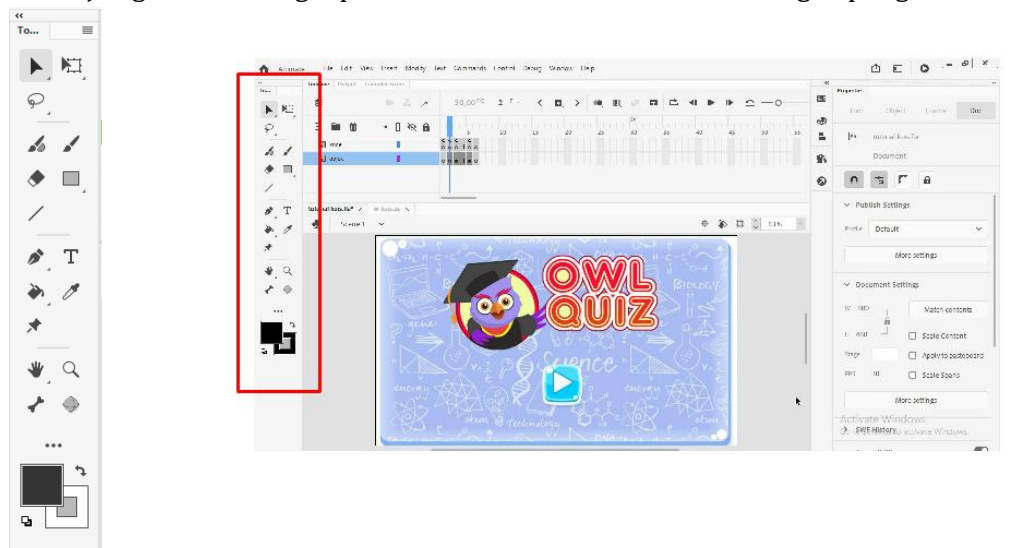
Menu utama merupakan fitur yang dapat ditemukan di sebagian besar aplikasi komputer. Terletak di bagian atas dan memiliki berbagai fungsi utama seperti pengaturan *file*, pengaturan aplikasi sampai dengan menu bantuan (*help*).



Gambar 73. Menu Utama

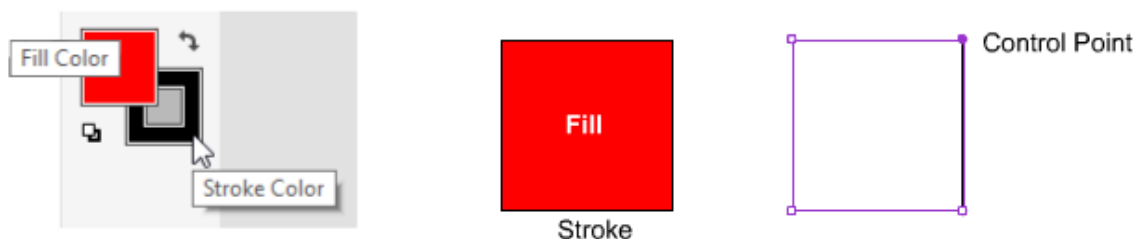
4.1.2 Tools

Tools merupakan perangkat standar aplikasi grafis yang berfungsi untuk menyeleksi objek, membuat objek gambar, menghapus atau melakukan fitur terkait dengan pengelolaan grafis.



Gambar 74. Tools

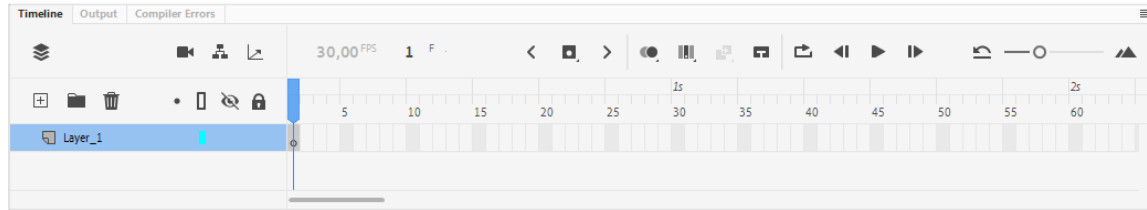
Animate sebagai aplikasi berbasis vektor memiliki dua elemen grafis yaitu garis tepi (*stroke*) dan warna isi (*fill*). Sebuah objek grafis bertipe vektor memiliki elemen fill, stroke atau keduanya.



Gambar 75. elemen grafis vektor

4.1.3 Timeline

Animate sebagaimana namanya merupakan aplikasi yang ditujukan untuk membuat animasi, olehkarena itu mutlak harus memiliki panel *Timeline*. *Timeline* memiliki urutan waktu yang disebut sebagai *Frame*. Untuk memahami *Timeline* dengan mudah, kita dapat membayangkan *Timeline* adalah lembaran-lembaran kertas yang digunakan untuk menyusun sebuah animasi.

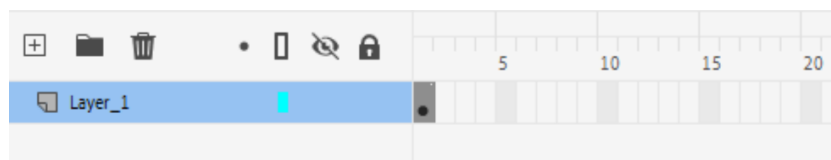


Gambar 76. *Timeline*

Sebuah *Timeline* dapat tersusun dari beberapa *Layer*. Selain *Layer* terdapat beberapa istilah yang harus dipahami terkait dengan *Timeline*, yang masing-masing ditampilkan dalam visual yang berbeda-beda, yaitu :

a. *Keyframe*

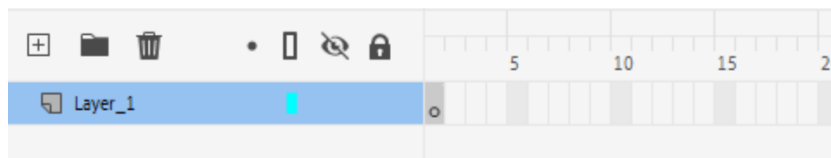
Keyframe atau *Frame* kunci merupakan sebuah titik di dalam *Timeline* yang memiliki objek. Dalam sebuah animasi titik *Keyframe* berfungsi sebagai titik awal atau titik akhir suatu gerakan, sedangkan dalam pengembangan aplikasi *Keyframe* menunjukkan keberadaan objek di suatu *Frame* tertentu. *Keyframe* divisualisasikan dengan titik hitam pada *Timeline*



Gambar 77. *Keyframe*

b. *Blank Keyframe*

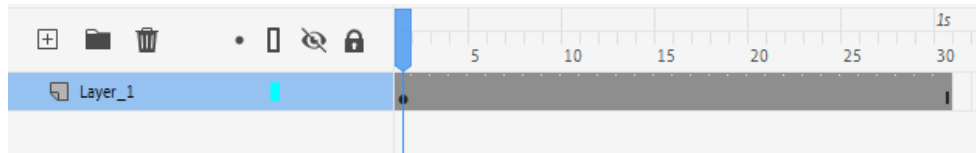
Berbeda dengan *Keyframe*, *Blank Keyframe* merupakan sebuah titik di dalam *Timeline* namun tidak memiliki objek atau kosong. *Blank Keyframe* divisualisasikan dengan titik kosong pada *Timeline*.



Gambar 78. *Blank Keyframe*

c. *Frame*

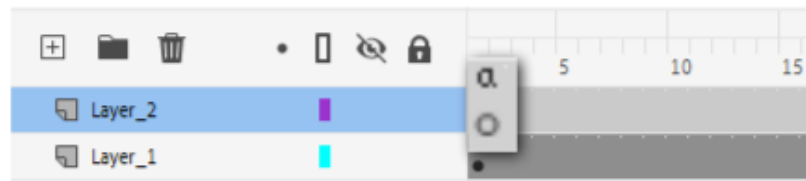
Frame merupakan konten *Timeline* yang memiliki durasi tertentu. Sebagai contoh, menampilkan sebuah gambar selama 1 detik, maka dibutuhkan *Frame* sejumlah 30 *Frame* (pada pengaturan kecepatan animasi 30 FPS).



Gambar 79. *Frame* berdurasi 1 detik

d. *Action*

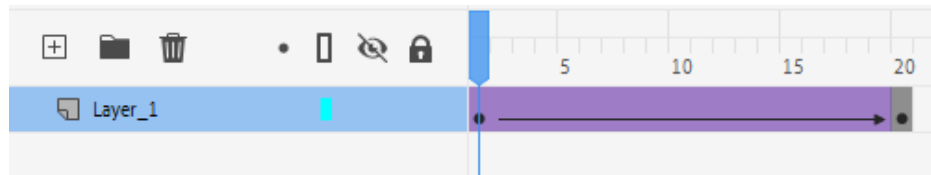
Kode dalam sebuah pemrograman aplikasi dapat ditempatkan di sebuah *Keyframe*. Hal ini akan ditampilkan dengan visualisasi huruf a kecil di atas titik *Keyframe*. *Keyframe* yang memiliki huruf a kecil menandakan di dalamnya terdapat kode.



Gambar 80. *Keyframe* dengan kode *Actionscript*

e. *Tween*

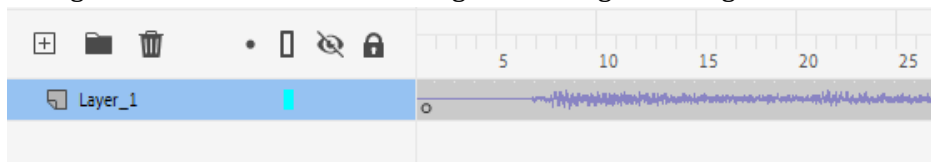
Tween merupakan gerakan objek yang terjadi di antara dua *Keyframe*. *Tween* ditandai dengan *Timeline* berwarna biru atau hijau.



Gambar 81. *motion tween*

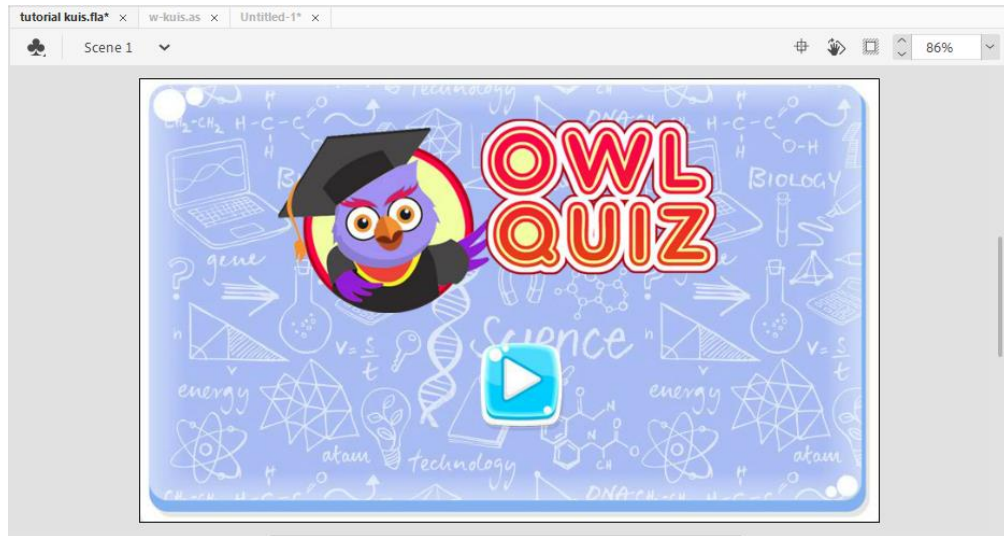
f. *Sound*

Frame dengan suara divisualisasikan dengan bentuk gelombang suara.



Gambar 82. *Frame* dengan suara

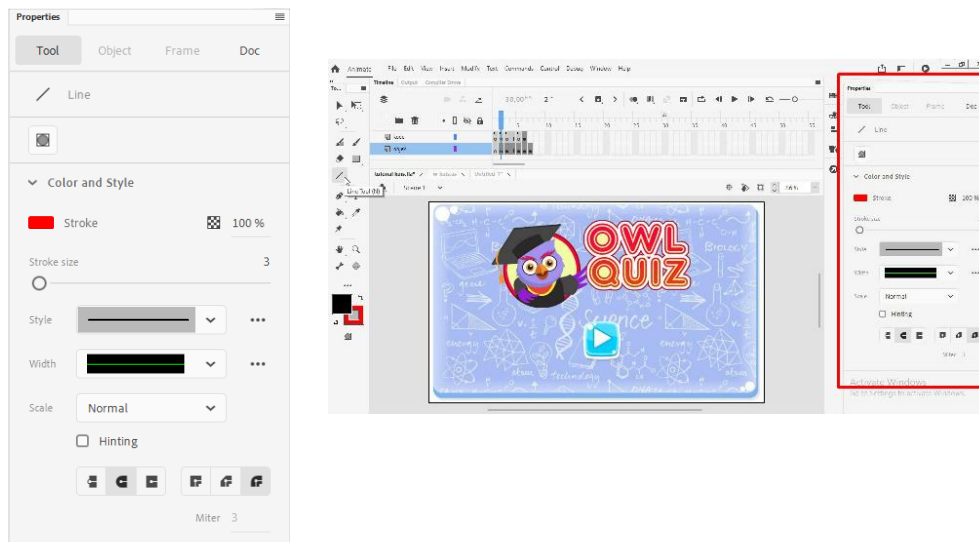
Stage merupakan tempat meletakkan objek dan batas area yang akan ditampilkan oleh aplikasi.



Gambar 83. *Stage*

4.1.5 Properties

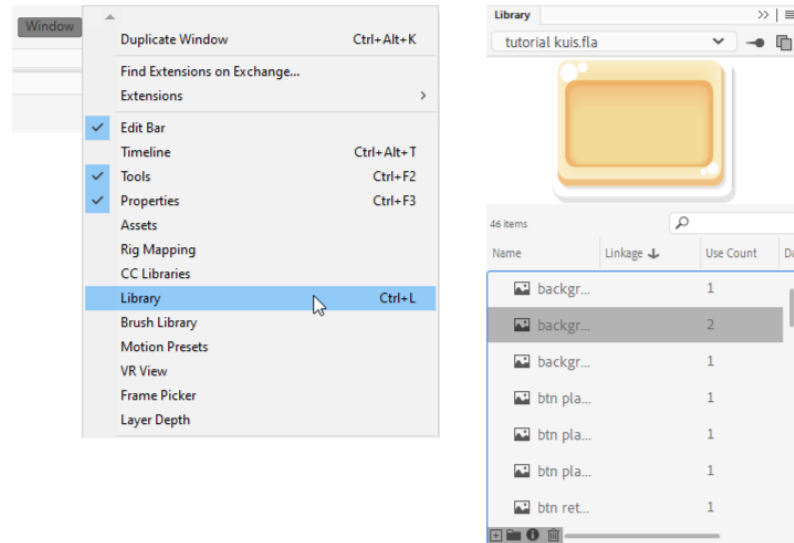
Properties merupakan panel dinamis yang selalu berubah-ubah seiring dengan *tool* yang sedang aktif. Panel *Properties* merupakan panel untuk menampilkan detail dari *tools* yang sedang digunakan, sebagai contoh ketika **Line Tool** aktif, maka panel *Properties* akan menampilkan secara detail pengaturan garis, seperti ukuran garis, jenis garis, warna garis dan sebagainya.



Gambar 84. panel *Properties*

4.1.6 Library

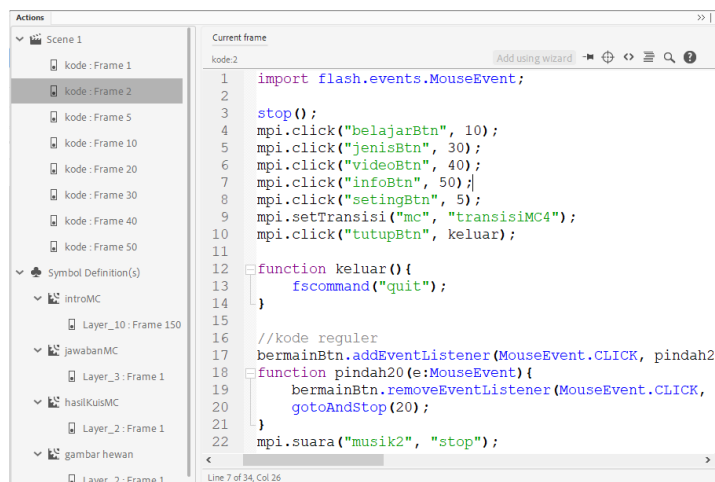
Library merupakan sebuah panel yang menyimpan seluruh aset yang digunakan di dalam proyek. *Library* akan menampung objek berupa gambar, *symbol*, suara dan elemen apapun yang digunakan di dalam aplikasi. *Library* dapat diakses dengan menekan menu **Window > Library** atau menekan tombol **Ctrl+L** atau menekan *icon* tombol *Library*.



Gambar 85. panel *Library*

4.1.7 Action Panel

Action panel adalah sebuah panel yang digunakan untuk menuliskan kode *Actionscript 3.0*. *Action* panel dapat diakses dengan menekan menu **Window > Action** atau dengan menekan tombol **F9**.



Gambar 86. panel *Action*

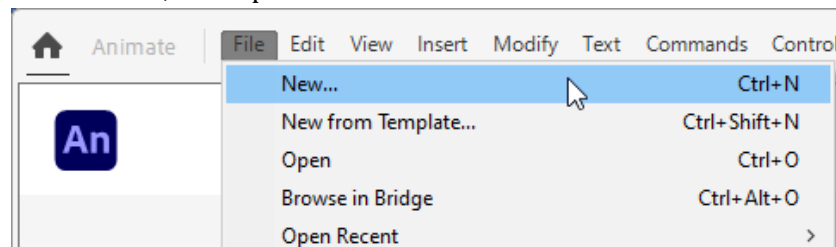
Pada dasarnya terdapat berbagai macam panel yang terdapat di area kerja Adobe Animate, namun demikian 7 elemen di atas yang paling sering diakses dalam mengembangkan aplikasi.

4.2 Membuat Proyek Baru dengan Adobe Animate

Setiap proyek yang dijelaskan di dalam buku ini akan diawali dengan proses pembuatan proyek baru. Buku ini menggunakan Adobe Animate 2021, namun tidak menutup kemungkinan tutorial yang ada di dalam buku ini dipraktikkan dengan menggunakan aplikasi Animate terbaru seperti versi 2022 ke atas, atau versi yang lebih lama seperti Adobe Animate 2017 atau Adobe Flash CC.

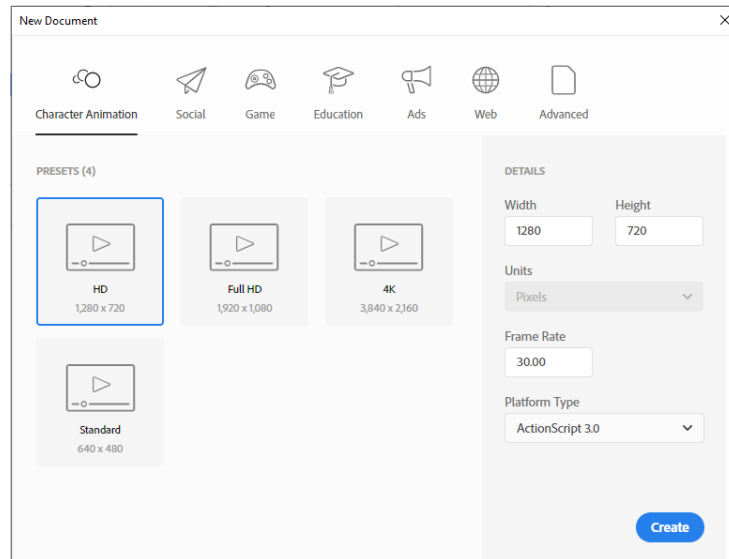
Untuk membuat sebuah proyek baru, perhatikan langkah berikut :

1. Buka aplikasi Adobe Animate
2. Klik menu **File > New**, maka panel dokumen baru akan muncul



Gambar 87. membuat *File* baru

3. Pada panel dokumen baru, terdapat beberapa pengaturan awal dari proyek yang dibuat. Animate versi 2017 ke atas, menyediakan beberapa *template* awal untuk proyek animasi, sosial media, iklan, *web* dan sebagainya. *Template* ini akan membedakan proyek dari 4 hal, yaitu :
 - a. **Width** yang merupakan lebar dari proyek dalam satuan *pixel*
 - b. **Height** yang merupakan tinggi dari proyek dalam satuan *pixel*
 - c. **Framerate** yang merupakan kecepatan sebuah animasi dijalankan dalam satuan *Frame Per Second* (FPS). Dalam bidang animasi FPS merupakan jumlah gambar yang digunakan untuk menyusun animasi berdurasi 1 detik, semakin banyak gambar yang digunakan animasi akan tampil semakin halus. Dalam konsep aplikasi, FPS akan menentukan seberapa cepat kode akan dieksekusi setiap detiknya. Secara *default* FPS bernilai 30, yang berarti kode akan dijalankan 30 kali dalam satu detik.
 - d. **Platform** merupakan luaran yang diharapkan dari proyek. Secara umum terdapat 3 jenis luaran, yaitu *Actionscript 3.0* yang merupakan luaran standar dari aplikasi berbasis Adobe Animate, *HTML 5 Canvas* untuk luaran *web based* yang dapat dioperasikan melalui *web browser*, dan *AIR* untuk luaran pada *platform* gawai Android atau iOS.



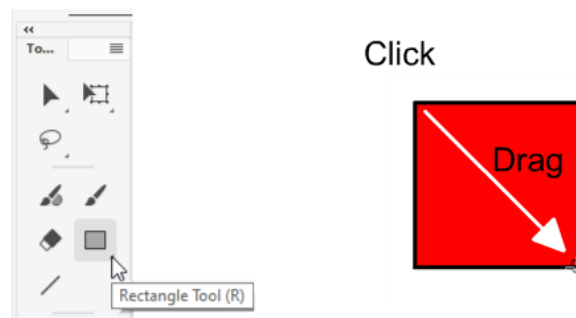
Gambar 88. panel *New Document*

4. Tekan tombol **Create**, maka beberapa saat kemudian dokumen baru akan terbentuk dan kita siap untuk memulai pembuatan aplikasi melalui Adobe Animate.

4.3 Objek dan *Symbol*

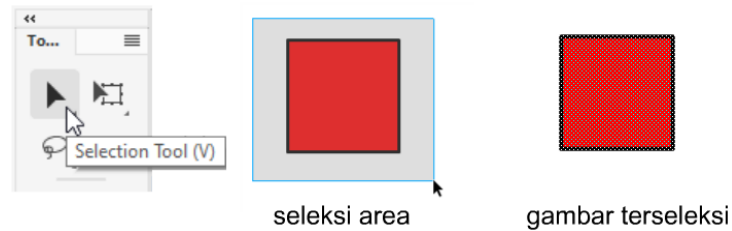
Dalam aplikasi Adobe Animate dikenal istilah objek dan simbol. Objek dan simbol merupakan segala aset yang digunakan di dalam aplikasi. Aset yang dimaksud disini adalah grafik, teks, suara, video atau elemen lainnya yang dapat diimpor dari luar aplikasi atau dibuat secara langsung menggunakan *tools* yang ada. Untuk memahami konsep tentang *symbol* perhatikan langkah berikut:

1. Buatlah sebuah *file* baru
2. Klik **Rectangle Tool**, lalu *drag* pada area *Stage* untuk membentuk sebuah gambar persegi. Perlu anda perhatikan bahwa persegi yang terbentuk memiliki 2 elemen, yaitu **stroke** (garis tepi) dan **fill** (warna isi).



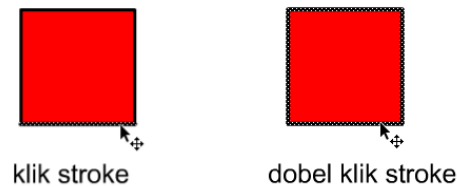
Gambar 89. membuat kotak

3. Setelah gambar persegi terbentuk, maka terdapat beberapa teknik untuk menyeleksinya. Apabila digunakan **Selection Tool** maka anda dapat menyeleksi gambar persegi tersebut dengan cara *drag* dan menyeleksi area persegi, jika seluruh area persegi terpilih maka persegi akan terseleksi ditandai dengan titik-titik di area persegi.



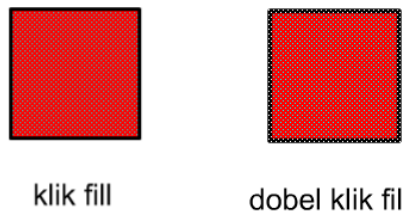
Gambar 90. menyeleksi gambar

Apabila digunakan **Selection Tool** dan **stroke** dari persegi tersebut diklik, maka hanya satu ruas **stroke** yang terpilih (gambar kiri). Apabila digunakan **Selection Tool** dan melakukan double klik pada **stroke** dari persegi tersebut maka seluruh **stroke** akan terpilih (gambar kanan).



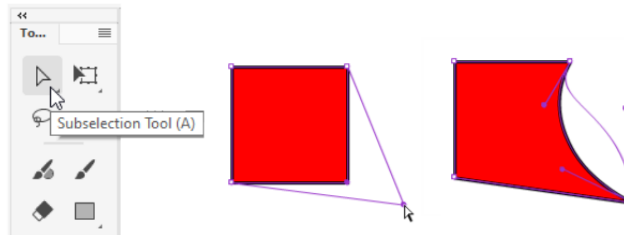
Gambar 91. menyeleksi *stroke*

Apabila digunakan **Selection Tool** dan anda klik **fill** dari persegi tersebut, maka hanya **fill** dari persegi saja yang terpilih (gambar kiri). Apabila digunakan **Selection Tool** dan anda double klik **fill** dari persegi tersebut maka **fill** dan seluruh **stroke** akan terpilih (gambar kanan).



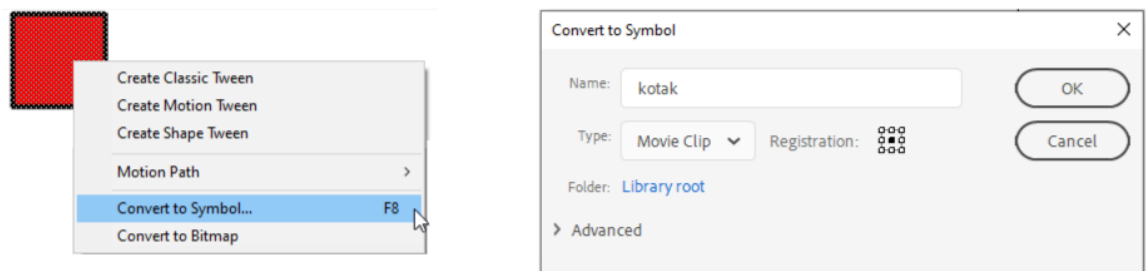
Gambar 92. menyeleksi *fill*

Apabila digunakan **Subselection Tool** dan anda klik **stroke** dari persegi tersebut, maka anda dapat mengedit titik-titik pembentuk persegi, dan anda dapat mengedit posisinya dengan cara *drag*.



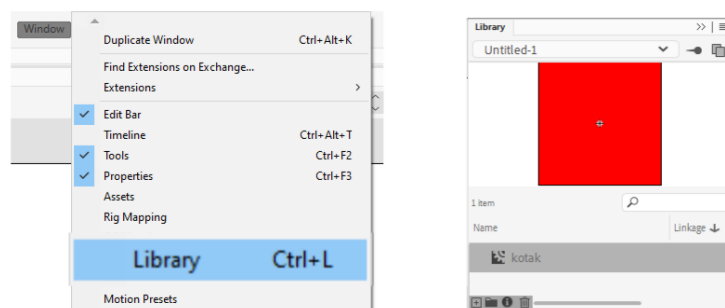
Gambar 93. mengedit titik dengan *subselection tool*

4. Untuk menjadikan kotak tersebut ke dalam bentuk *symbol*, seleksi seluruh kotak. Klik kanan dan pilih opsi **Convert to Symbol** atau menekan tombol **F8**. Maka panel *Convert to symbol* akan muncul. Pada panel tersebut terdapat beberapa opsi yang perlu diperhatikan :
 - a. **Name** merupakan nama *symbol*, sebisa mungkin ketikkan nama yang spesifik agar mudah untuk diidentifikasi
 - b. **Type** merupakan tipe dari *symbol* yaitu *Movie Clip*, *Button* dan *Graphic*.
 - c. **Registration** merupakan titik pusat dari *symbol* yang menunjukkan posisi kordinat sumbu x dan y ketika diidentifikasi oleh kode.
 - d. **Advanced** merupakan panel tambahan yang digunakan untuk pengaturan *symbol* terkait identifikasi *symbol* terhadap kode *Actionscript*



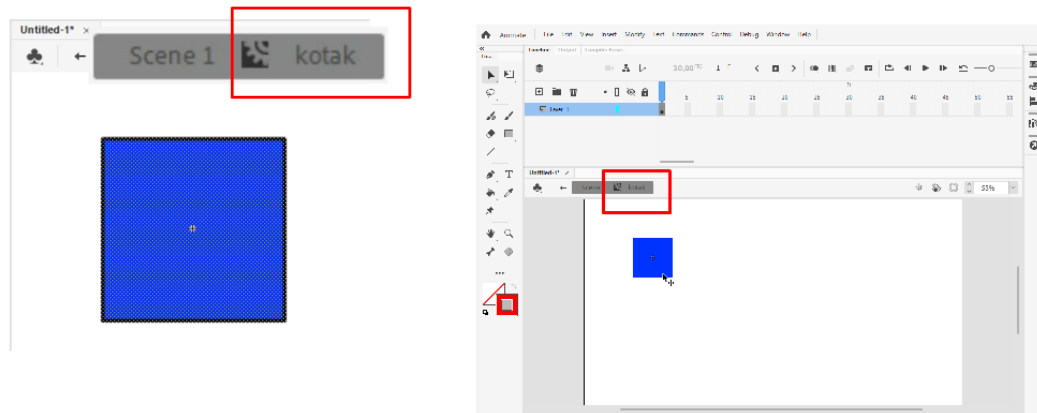
Gambar 94. *Convert to Symbol*

5. Untuk belajar di tahap pertama, ketikkan nama “kotak”, tipe *MovieClip* dan titik registrasi di tengah. Klik OK, maka *symbol* akan terbentuk. Setiap *symbol* yang terbentuk akan disimpan ke dalam **Library**. *Library* dapat diakses dengan menekan **Ctrl + L**, menekan icon **Library** atau menekan menu **Window > Library**.



Gambar 95. *Symbol* Kotak pada panel *Library*

6. Pada saat sebuah objek grafis di*convert* menjadi sebuah *symbol*, maka grafis tidak dapat lagi diedit secara langsung pada *Stage* utama. Sebagai contoh untuk mengubah warna kotak yang telah di*convert* menjadi *MovieClip* tersebut harus masuk ke mode edit *symbol*. Untuk masuk ke mode edit *symbol* kita dapat melakukan **dobel klik pada *symbol***, atau menyeleksi *symbol* dan menekan tombol **Ctrl+E**. Setelah masuk ke mode edit *symbol*, maka kotak akan dapat diubah bentuk maupun warnanya dengan *drawing tools*.



Gambar 96. mode edit *symbol*

7. Untuk kembali ke *Stage* utama (*Scene* utama), tekan **Ctrl+E** atau tekan teks ***Scene 1***.

Pada tahapan ini diketahui cara mengubah atau meng*convert* sebuah objek grafis menjadi sebuah *symbol*. Grafis yang dimaksud tidak harus dibuat dengan menggunakan aplikasi Adobe Animate, namun juga dapat digunakan *software* pengelola grafis lainnya seperti Adobe Photoshop, Illustrator, Corel dan sebagainya. Proses mengimpor grafis dari luar aplikasi Animate akan dijelaskan pada bab berikutnya.

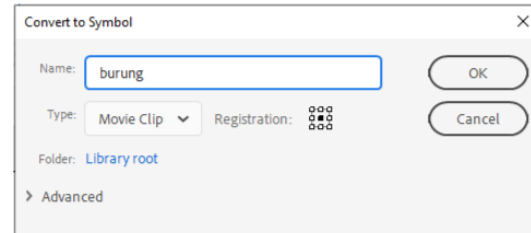
4.3.1 *MovieClip*

MovieClip adalah sebuah simbol yang memiliki *multi Frame* secara independen dan dapat diputar melalui sebuah *Frame* di *Timeline* utama. Bayangkan kita akan membuat sebuah animasi burung yang sedang terbang dari kiri ke kanan. Selain bergerak dari kiri ke kanan, burung juga memiliki gerakan mengepakkan sayap. Untuk membuat animasi tersebut akan lebih mudah jika digunakan *symbol* bertipe *MovieClip*. Animasi burung yang sedang mengepakkan sayap bisa kita buat terlebih dahulu dalam format *MovieClip*, selanjutnya barulah kita gerakkan di *Timeline* utama agar bisa terbang dari kiri ke kanan.

MovieClip menjadi *symbol* multifungsi pada aplikasi Animate karena dapat dengan mudah diatur dengan kode *Actionscript*. *Timeline* yang independen di dalam *movieclip* juga mempermudah pengaturan interaktivitas yang kompleks.

Untuk lebih memahami konsep *MovieClip*, perhatikan contoh berikut :

1. Buatlah sebuah *file* baru (**tutorial 1 – *MovieClip***)
2. Dengan menggunakan *drawing tool*, gambarlah sebuah gambar burung yang sederhana seperti pada gambar berikut :

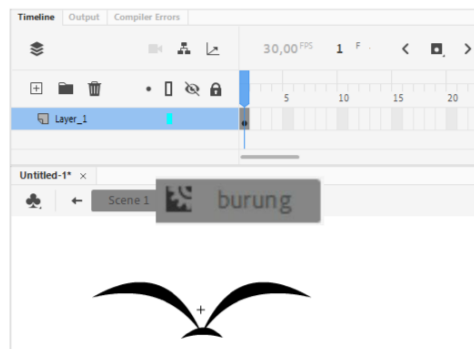


Gambar 97. membuat *MovieClip* burung

3. Klik kanan kemudian pilih **Convert to Symbol**. Ketikkan “**burung**” pada kolom nama dan tekan **OK**.

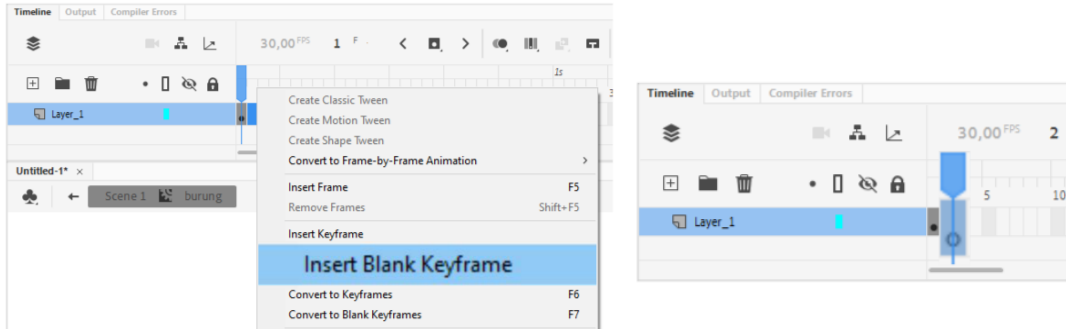
Note : Pada tahapan selanjutnya di dalam buku ini, langkah ini akan ditulis dengan bahasa yang lebih singkat yaitu “buatlah sebuah *MovieClip* dengan nama burung”.

4. **Dobelklik** *MovieClip* burung untuk masuk ke mode edit. Maka pada saat ini kita berada di dalam *MovieClip* burung, *Timeline* yang ada adalah milik *Timeline* burung, sehingga apapun yang kita lakukan terhadap *Timeline* hanya akan berimplikasi pada *MovieClip* burung, dan tidak mempengaruhi *Timeline* utama.



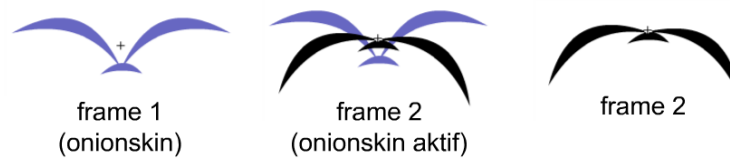
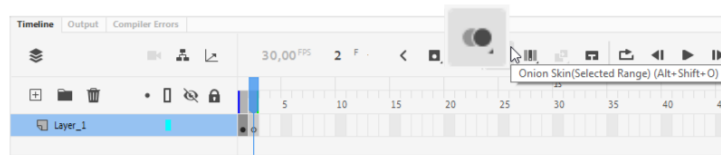
Gambar 98. mode edit *MovieClip* burung

5. Klik kanan **Frame 2** kemudian pilih opsi **Insert Blank Keyframe**. Maka akan muncul lingkaran kosong pada *Timeline* dan gambar burung akan menghilang. Menambahkan *Blank Keyframe* pada *Timeline* dapat diibaratkan sebagai penambahan kertas kosong pada pembuatan animasi. Pada lembar kertas pertama (*Frame 1*) sudah terdapat animasi burung yang mengepakkan sayapnya ke arah atas, kemudian pada lembar ke dua yang masih kosong (*Frame 2*), kita perlu menambahkan sebuah gambar burung yang kepakannya ke arah bawah.



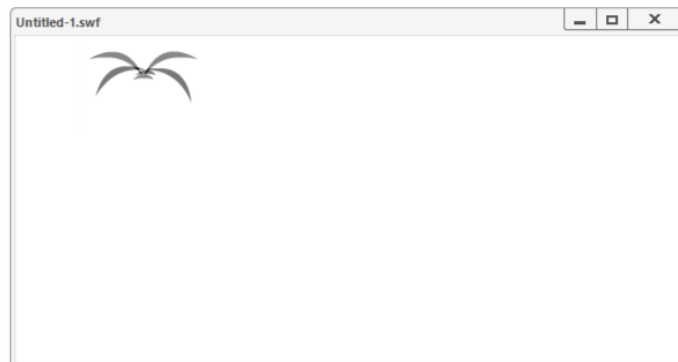
Gambar 99. menambahkan *Blank Keyframe*

6. Selanjutnya gambarkan burung dengan posisi sayap yang berbeda (ke arah bawah). Untuk mempermudah proses tersebut kita dapat menggunakan fitur **onion skin**. Fitur ini akan menunjukkan gambar yang berada di *Frame* sebelumnya, sehingga dapat digunakan sebagai acuan.



Gambar 100. membuat gambar pada *Frame 2*

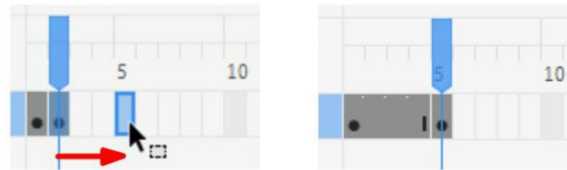
7. Jalankan aplikasi dengan menekan tombol **Ctrl+Enter**. Maka akan muncul sebuah panel yang menampilkan animasi secara berulang (panel SWF). Pada aplikasi Adobe Animate, **Ctrl+Enter** merupakan *shortcut* dari **test Movie** yang bertujuan untuk melihat luaran aplikasi yang dibuat. Panel ini akan menampilkan *file* bertipe SWF, *file* animasi standar yang dihasilkan oleh aplikasi Adobe Animate, Adobe Flash, dan Macromedia Flash (yang merupakan satu kesatuan *brand/merk*).



Gambar 101. luaran *MovieClip* burung

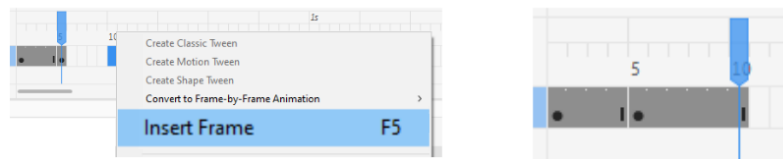
Namun apabila anda lihat, animasi burung yang muncul berkedip-kedip. Hal ini dikarenakan kecepatan animasi yang tinggi yaitu 30 FPS, sementara animasi yang dibuat hanya memiliki durasi 2 *Frame* saja (jika dihitung berarti hanya 2/30 detik). Oleh karena itu diperlukan pengaturan durasi animasi agar gerakan burung terlihat dengan baik.

8. Klik *Keyframe* 2, kemudian **drag** menuju ke *Frame* 5. Hal ini dimaksudkan untuk memberikan durasi gambar di *Frame* 1 sebanyak 5 *Frame* (5/30 detik).



Gambar 102. menggeser *Frame*

9. *Keyframe* yang kedua (*Frame* 5) juga memerlukan durasi. Oleh karena itu klik kanan *Frame* 10 dan pilih opsi **insert *Frame***.



Gambar 103. menambahkan durasi dengan *insert Frame*

10. Tekan **Ctrl+E** untuk kembali ke *Scene* utama, lalu jalankan kembali aplikasi dengan menekan **Ctrl+Enter**. Kali ini animasi burung sudah dapat dilihat dengan baik.

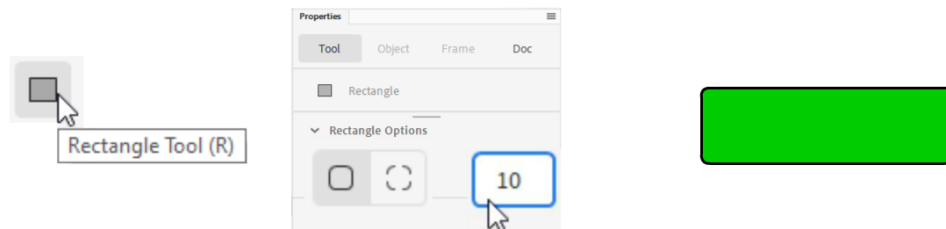
Perhatikan bahwa *MovieClip* diputar secara terus menerus meskipun pada *Scene* utama *MovieClip* ditampilkan dalam satu *Frame*. *MovieClip* burung ini nantinya dapat kita animasikan kembali, sebagai contoh dianimasikan bergerak dari kiri ke kanan, maka selain burung bergerak dari kiri ke kanan, animasi mengepakkan sayap akan terus dijalankan, sehingga didapatkan animasi burung mengepakkan sayap dan bergerak dari kiri ke kanan (lihat tutorial pada buku [Desain dan Pemrograman Multimedia Interaktif](#) pada bab animasi).

4.3.2 Tombol

Tombol adalah sebuah elemen interaktif yang mengharuskan pengguna untuk menekan *mouse* atau menyentuh tombol untuk mengaktifkan suatu perintah tertentu. Dalam sebuah pengembangan aplikasi keberadaan tombol selalu dibutuhkan untuk mendapatkan *input* dari pengguna. Elemen utama pada *symbol* bertipe tombol yang membedakan dengan *symbol* lain adalah jumlah *Frame* pada *Timelinenya* yang hanya 4 buah *Frame*, di luar hal tersebut proses pembuatannya hampir sama dengan pembuatan *MovieClip*.

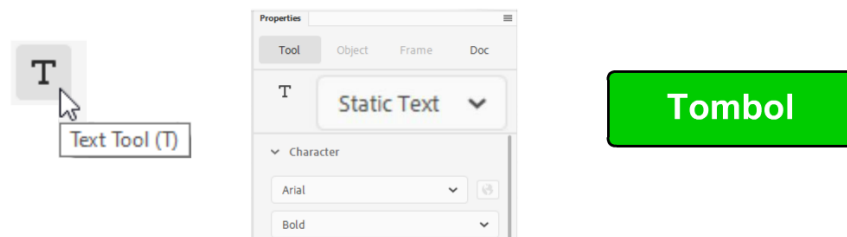
Agar lebih jelas memahami tentang tombol, ikuti langkah berikut :

1. Buatlah sebuah *file* baru (**tutorial 2 – Tombol**)
2. Buatlah sebuah kotak dengan ujung membulat (*rounded rectangle*). Klik **rectangle tool** kemudian perhatikan di panel **Properties**. Pada panel tersebut terdapat opsi **Rectangle Option**, yang merupakan radius dari lengkungan di ujung kotak. Ketikkan angka **10** pada kolom tersebut untuk menghasilkan kotak dengan ujung melengkung beradius 10 *pixel*.



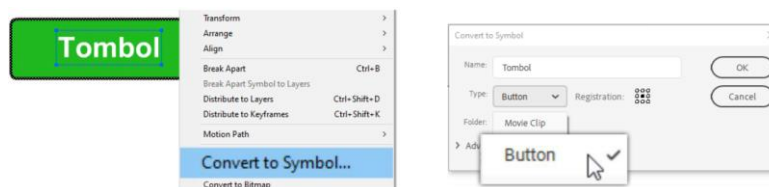
Gambar 104. membuat kotak

3. Selanjutnya klik **Text tool** untuk membuat teks. Pastikan opsi yang terpilih di panel **Properties** adalah teks bertipe **Static Text**. Atur jenis huruf, ukuran dan warna huruf sesuai keinginan kemudian klik di atas kotak. Ketikkan “Tombol” lalu klik di luar area teks untuk menonaktifkan seleksi.



Gambar 105. menambahkan *Static Text*

4. Drag seluruh area kotak dan teks untuk menyeleksi semuanya. Klik kanan kemudian pilih opsi **Convert to Symbol**. Pada panel tersebut ketikkan nama “tombol” dan pilih tipe **Button** lalu tekan tombol **OK**



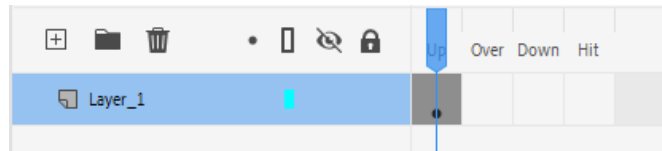
Gambar 106. membuat *symbol* bertipe *button*

5. Jalankan aplikasi dengan menekan **Ctrl+Enter**. Dekatkan *mouse* ke tombol maka kursor *mouse* akan berubah menjadi bentuk tangan, menandakan bahwa tombol telah aktif.



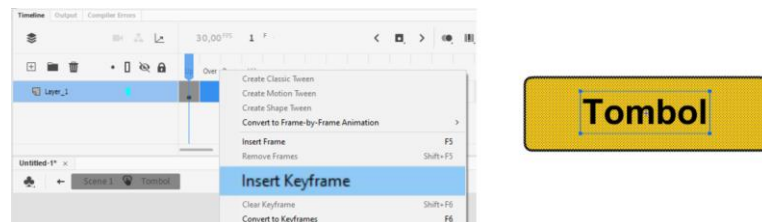
Gambar 107. kursor *mouse* di atas tombol

6. Kembali ke proyek, kemudian edit tombol dengan cara **dobel klik tombol**. Pada mode edit *symbol* bertipe *button* akan terlihat 4 buah *Frame*, yaitu *Frame up*, *over*, *down* dan *hit*. *Up* adalah visualisasi tombol ketika berstatus normal, *over* adalah ketika *mouse* melintas di atas area tombol, *down* adalah ketika tombol ditekan atau disentuh, dan *hit* adalah area sensor tombol. Pada saat tombol baru dibuat, maka hanya *Frame up* yang memiliki *Keyframe*.



Gambar 108. struktur *Frame symbol* bertipe *button*

7. Untuk menambahkan visualisasi pada *Frame over*, klik kanan *Frame over* dan pilih opsi **Insert Keyframe**. Ubah warna kotak menjadi warna oranye dan warna teks menjadi hitam.



Gambar 109. menambahkan *Frame Over* pada tombol

8. Jalankan kembali aplikasi dengan menekan **Ctrl+Enter**, maka ketika *mouse* melintas di atas tombol, akan terjadi perubahan warna. Pada tahapan ini efek yang dapat diletakkan pada *Frame over* tidak hanya perubahan warna saja. Kita dapat meletakkan beberapa efek atau meletakkan *MovieClip* ke *Frame over* tersebut. Sebagai contoh apabila ingin menambahkan efek tombol berkilau ketika *mouse* melintas di atasnya, atau efek lainnya.

Untuk menambahkan gambar di *Frame down*, dapat dilakukan langkah yang sama. Visualisasi pada *Frame down* akan ditampilkan saat tombol ditekan atau disentuh. Sementara untuk *Frame hit* kita dapat mengabaikannya karena visualisasi pada *Frame hit* tidak akan ditampilkan. *Hit* merupakan area sensor dari tombol, sebagai contoh apabila tombol kita terlalu kecil, maka kita dapat menambahkan objek grafik berukuran lebih besar pada *Frame hit*. Pada tahapan ini tombol selesai dibuat. Pada tahapan selanjutnya proses pembuatan tombol akan dituliskan dalam bahasa yang lebih singkat seperti “buatlah sebuah tombol”.

4.3.3 Graphic

Graphic memiliki konsep yang hampir sama dengan *MovieClip*, yaitu memiliki *Timeline* yang independen. Namun perbedaan utama dengan *MovieClip* adalah adanya kemampuan untuk mengatur *Frame* yang ditampilkan. *MovieClip* pada contoh pembuatan animasi burung di atas, akan menampilkan burung dimulai dari *Frame* 1 dan bergerak secara *looping* (terus menerus). Sementara *Graphic* dapat diatur mulai dimunculkan dari *Frame* berapapun.

Perbedaan berikutnya adalah untuk menampilkan *Graphic*, diperlukan durasi pada *Timeline* utama. Pada *MovieClip* burung di atas, dapat ditampilkan pada 1 buah *Frame* saja. Sementara pada *symbol* bertipe *Graphic* dibutuhkan durasi yang lebih panjang untuk menampilkannya.

Perbedaan berikutnya adalah *Graphic* tidak dapat diberikan *instance name* yang artinya *Graphic* tidak dapat diakses oleh kode *Actionscript*. Sehingga penggunaan *Graphic* adalah khusus untuk pengembangan animasi saja, tidak dapat diatur secara dinamis dengan kode sebagaimana *MovieClip*.

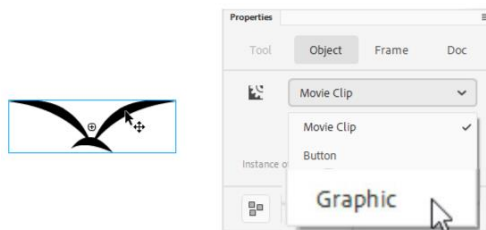
Untuk lebih memahami tentang *Graphic*, perhatikan contoh berikut :

1. Buka kembali *file* animasi burung, atau buat terlebih dahulu *MovieClip* burung jika belum membuatnya. (**tutorial 3 – Graphic**)
2. Pada layar sudah terdapat sebuah *MovieClip* burung. Seleksi *MovieClip* burung kemudian **copy-paste** untuk menambahkan *MovieClip* burung di layar (pada tahapan ini di layar terdapat 2 *MovieClip* burung)



Gambar 110. hasil *copypaste* *MovieClip* burung

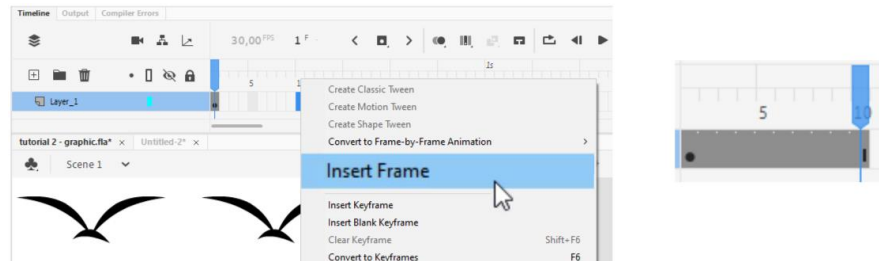
3. Seleksi burung yang baru saja dipaste, kemudian pada panel **Properties**, ubah tipe *symbol* menjadi **Graphic**.



Gambar 111. mengubah tipe menjadi *Graphic*

4. Jalankan aplikasi dengan menekan **Ctrl+Enter**. Saat ini akan terlihat bahwa burung ke 1 (*MovieClip*) dapat bergerak (beranimasi), sedangkan burung 2 (*Graphic*) hanya diam saja.

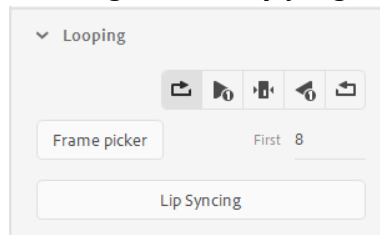
5. Klik kanan **Frame 10** pada *Timeline* kemudian pilih opsi **insert Frame**. Hal ini dimaksudkan untuk memberikan durasi animasi sepanjang 10 *Frame* (10/30 detik). Jalankan kembali aplikasi dengan menekan **Ctrl+Enter**, maka kedua burung akan bergerak (beranimasi).



Gambar 112. menambahkan durasi animasi

Sebuah *symbol* bertipe *Graphic* membutuhkan durasi yang lebih panjang untuk menjalankan animasi di dalamnya. Berbeda dengan *symbol* bertipe *MovieClip* yang dapat bergerak meskipun ditampilkan dalam 1 *Frame* saja.

6. Keunggulan *symbol* bertipe *Graphic* adalah adanya opsi pengaturan *Frame*. Klik **Graphic** burung, kemudian lihat pada panel **Properties** di bagian **Looping**. Terdapat 5 buah opsi yang dapat dipilih, yaitu memainkan *Graphic* secara *loop* (berulang terus menerus), memainkan *Graphic* satu kali putaran saja, menampilkan 1 *Frame* tertentu, memainkan secara mundur (*reverse*) sebanyak satu kali, dan menampilkan mundur secara berulang (*reverse loop*). Pada bagian **first** juga dapat diatur, pada *Frame* berapa animasi mulai diputar, berbeda dengan *MovieClip* yang selalu diputar dari *Frame* 1.

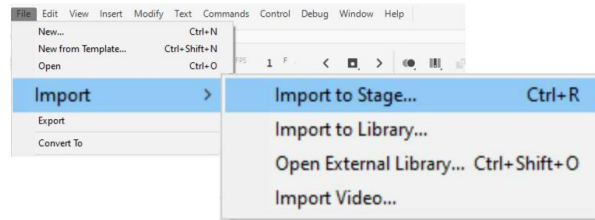


Gambar 113. pengaturan *looping* pada *Graphic*

4.3.4 Bitmap

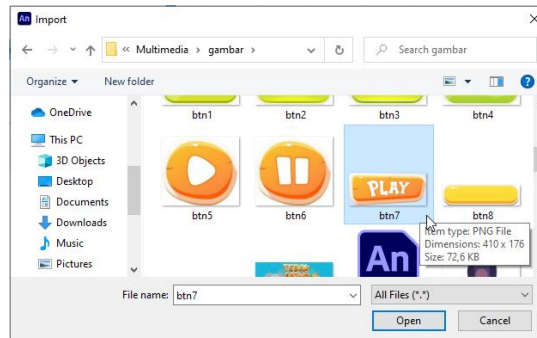
Bitmap merupakan objek yang berasal dari proses import atau copy paste. Untuk mengimport objek bertipe *Bitmap*, cukup dengan memilih menu **File > Import > Import to Stage**, maka opsi pilihan import akan muncul. Animate dapat mengimport grafik berbagai tipe *file*, namun untuk pengembangan aplikasi multimedia interaktif disarankan untuk menggunakan gambar bertipe **JPG** atau **PNG**. Perhatikan contoh berikut :

1. Buatlah sebuah *file* baru (**tutorial 4 – Import File**)
2. Pilih menu **File > Import > Import to Stage**



Gambar 114. mengimport file

3. Pilih file gambar (*Bitmap*), sebagai contoh dipilih file gambar untuk tombol *play*



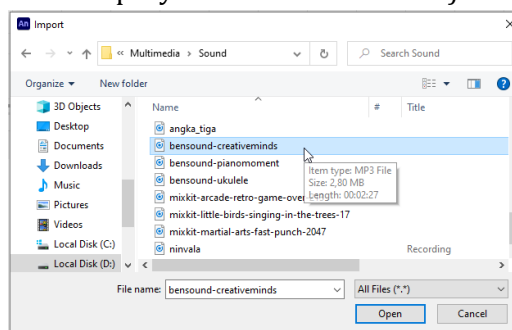
Gambar 115. memilih gambar yang diimport

4. Objek yang diimport akan berada di *Stage* dan secara otomatis akan masuk ke dalam *Library*.

4.3.5 Suara

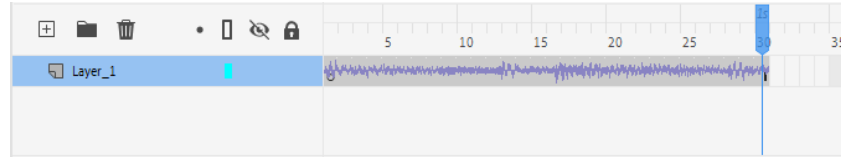
Seperti halnya *Bitmap*, objek bertipe suara juga dapat diimport secara langsung ke *Stage* maupun ke *Library*. Berbeda dengan objek bertipe visual (teks, grafis, *Bitmap*), suara tidak memiliki wujud yang bisa diseleksi langsung dari *Stage*. Namun suara ditampilkan dalam bentuk gelombang pada *Timeline*. Untuk memahami lebih lanjut, ikuti langkah berikut :

1. Buatlah sebuah file baru (**tutorial 5 – Suara**).
2. Importlah sebuah file suara ke *Stage* dengan memilih opsi **File > Import to Stage**. File suara yang dapat diimport adalah file bertipe MP3, WAV, SND dan sebagainya. Namun yang direkomendasikan untuk proyek multimedia adalah file bertipe WAV dan MP3.



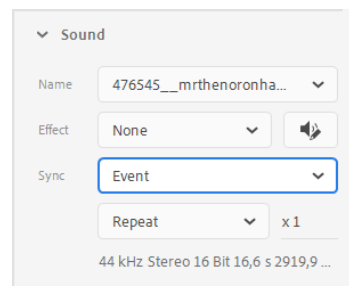
Gambar 116. mengimpor suara

3. Untuk memastikan suara telah diimpor dan berada di *Stage*, dapat dilihat dari *Frame* tempat suara tersebut diletakkan. Dalam contoh ini berarti suara terletak di *Frame* 1. Akan terlihat sebuah gelombang pada *Keyframe*. Jalankan aplikasi dengan menekan **Ctrl+Enter**. Maka akan terdengar sebuah suara yang telah diimport ke *Stage*.
4. Untuk memahami lebih lanjut terkait dengan peletakan suara di sebuah *Frame*, klik kanan **Frame 30** dan pilih opsi **Insert Frame**. Apabila durasi suara yang diimport lebih dari 1 detik, maka tampilan suara pada *Timeline* akan membentuk sebuah gelombang.



Gambar 117. tampilan gelombang suara pada *Timeline*

5. Jalankan kembali aplikasi dengan menekan **Ctrl+Enter** dan suara akan terdengar, namun terdapat kemungkinan suara saling menumpuk satu sama lain. Hal tersebut dikarenakan aplikasi yang berjalan secara *looping*. Suara yang berjalan mulai dari *Frame* 1, dan apabila durasi suara lebih dari 1 detik, maka saat aplikasi berulang kembali ke *Frame* 1 maka suara baru akan dimainkan sementara suara yang telah dimainkan sebelumnya belum selesai, sehingga terjadilah penumpukan suara. Oleh karena itu terdapat pengaturan khusus agar suara tidak saling bertumpuk.
6. Klik *Frame* 1 lalu lihat panel **Properties**. Pada opsi **Sound** terdapat pengaturan yang lebih kompleks untuk mendapatkan efek pemutaran suara yang berbeda-beda, yaitu pada opsi **Sync**.



Gambar 118. *Properties* suara

- a. **Event** akan memainkan suara setiap kali *Frame* aktif melewati sebuah *Keyframe* yang memiliki suara. Dalam contoh ini secara *default* seting *sync* adalah *event*, sehingga suara baru akan dimainkan setiap kali melewati *Frame* 1, sehingga suara bertumpuk.
- b. **Start** akan memainkan suara baru setiap kali *Frame* aktif melewati *Keyframe* dan memutar suara tersebut sampai selesai. Sehingga meskipun suara berdurasi lebih dari 1 detik, pada contoh ini apabila memilih opsi *start*, akan memainkan suara sampai selesai dan ketika melewati *frame* 1 kembali akan memainkan ulang suara. Pada opsi ini tidak akan terjadi penumpukan suara.

- c. **Stop** digunakan untuk menghentikan suara. Suara akan mati ketika melewati *Keyframe* dengan opsi *stop*.
- d. **Stream** akan memainkan suara sesuai dengan durasi yang dimiliki oleh *Frame* suara. Pada contoh ini misalnya, suara akan dimainkan selama 1 detik, kemudian akan diulangi lagi dari awal.

Opsi *sync* tersebut dapat digunakan sesuai dengan kebutuhan, atau pada tahapan selanjutnya pengaturan suara akan menjadi lebih fleksibel dengan menggunakan kode *Actionscript*.

4.3.6 Video

Video sebagai objek di dalam aplikasi Adobe Animate dapat dimpor secara internal maupun secara eksternal. Video memiliki opsi khusus yang akan dijelaskan pada bab selanjutnya.

4.4 Animasi

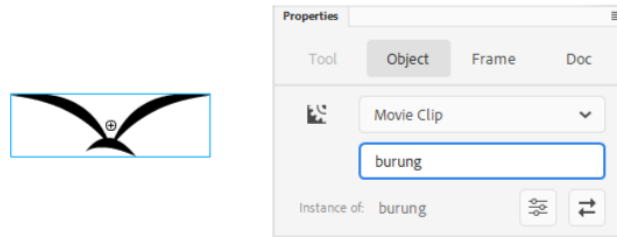
Adobe Animate, sesuai dengan suku kata pembentuk namanya yaitu *Animate* secara spesifik digunakan untuk membuat proyek animasi, sehingga memiliki fitur-fitur khusus yang mendukung animasi. Animasi dengan menggunakan teknik *motion tween* dan *motion guide* telah dijelaskan pada buku sebelumnya, sehingga tidak dijelaskan secara spesifik dalam buku ini.

4.5 Instance name

Instance name adalah sebuah nama yang diberikan kepada *symbol* yang berada di *Stage*. Ketika sebuah objek diubah menjadi sebuah *symbol* dan berada di *Stage*, maka objek tersebut disebut sebagai *instance*. Di sebuah *Stage* dapat diisi dengan satu atau lebih *symbol* yang sama, sebagai contoh ketika sebuah *MovieClip* “burung” telah dibuat lalu diletakkan ke *Stage*, maka “burung” dapat *dicopy paste* atau diletakkan berapapun jumlahnya di *Stage*. *MovieClip* “burung” yang berada di *Stage* tersebut disebut sebagai *instance*.

Dalam sebuah pemrograman aplikasi, khususnya multimedia interaktif, maka diperlukan identifikasi terhadap masing-masing *instance* yang akan diakses oleh kode. Identifikasi tersebut disebut dengan ***instance name***. Untuk lebih memahami tentang *instance name*, perhatikan langkah berikut :

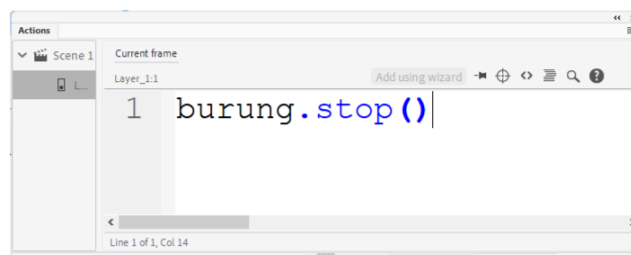
1. Buatlah sebuah *file* baru (**tutorial 6 – Instance name**)
2. Buatlah sebuah *MovieClip* “burung” yang di dalamnya terdapat animasi kepakan sayap burung (lihat tutorial 1).
3. Klik *MovieClip* “burung”, kemudian pada panel **Properties** terdapat kolom ***instance name***. Ketikkan “burung” pada kolom *instance name* tersebut.



Gambar 119. menambahkan *instance name*

4. Klik **Frame 1**, kemudian buka panel **Action** dengan menekan menu **Window > Action**. Pada panel **Action** ketikkan kode berikut :

```
burung.stop();
```



Gambar 120. menambahkan kode

5. Jalankan aplikasi dengan menekan tombol **Ctrl+Enter**. Maka *MovieClip* burung akan diam (berhenti). Pada kode tersebut “burung.stop()” mengacu pada 2 hal, “burung” adalah nama instance dari *MovieClip* burung, dan stop() adalah perintah untuk menghentikan *Frame*.
6. Apabila *instance name* belum ditulis, atau salah ketik maka akan muncul pesan error.



Gambar 121. pesan error ketika terjadi kesalahan *instance name*

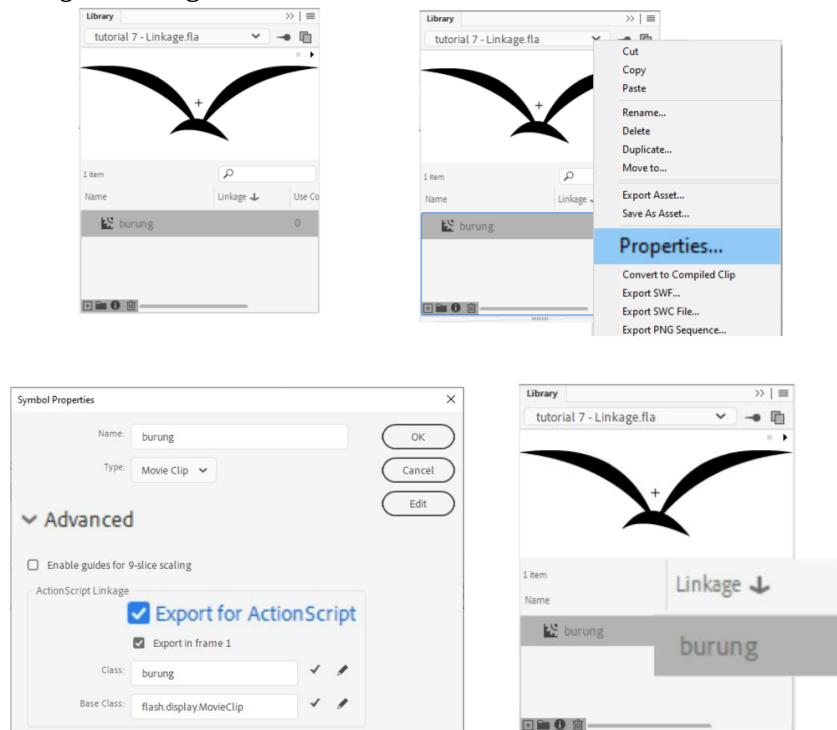
4.6 Linkage

Seperti halnya *instance name*, *Linkage* adalah sebuah identitas yang dimiliki oleh *symbol*, akan tetapi *Linkage* tidak melekat pada *symbol* yang berada di *Stage (instance)*, akan tetapi *Linkage* dapat dilihat pada panel *Library*. Sebuah *symbol* yang memiliki *Linkage* dapat diletakkan ke *Stage* melalui kode.

Perbedaan konsep yang mudah dipahami antara *Linkage* dan *instance name* adalah, *instance name* diberikan kepada *symbol* yang telah berada di *Stage*, sementara *Linkage* adalah meletakkan *symbol* dari *Library* menuju ke *Stage* melalui kode. Perhatikan contoh berikut :

1. Buatlah sebuah *file* baru (**tutorial 7 – Linkage**)
2. Buatlah sebuah *MovieClip* “burung”
3. Setelah *MovieClip* burung terbentuk, seleksi kemudian hapus dari *Stage* dengan menekan tombol **delete**. Sehingga tidak ada *symbol* apapun di *Stage* saat ini
4. Buka panel **Library** dengan menekan opsi **Window > Library** atau menekan **Ctrl+L** atau menekan icon *Library*. Selanjutnya **klik kanan** *MovieClip* burung dan pilih opsi **Properties**.

Pada panel **Symbol Properties** klik tab **Advanced** untuk menampilkan opsi tambahan. Centang opsi **Export for Actionscript**, maka pada kolom **Class** akan muncul nama **burung**, inilah nama *Linkage* yang akan muncul. Klik **OK**, maka pada panel *Library* akan terlihat *Linkage* “burung”.



Gambar 122. menambahkan *Linkage*

5. Klik **Frame 1**, kemudian buka panel **Action** dengan menekan menu **Window > Action**. Pada panel *Action* ketikkan kode berikut :

```
var birdMC = new burung;
addChild(birdMC);
birdMC.x = 100;
birdMC.y = 200;
```

6. Jalankan dengan menekan tombol **Ctrl+Enter**, maka *MovieClip* burung akan muncul di layar.

Dengan menggunakan *Linkage* kita dapat menambahkan *symbol* yang berada di dalam *Library* ke *Stage* dengan pengaturan-pengaturan kode *Actionscript* yang akan dijelaskan pada bab berikutnya.

BAB 5

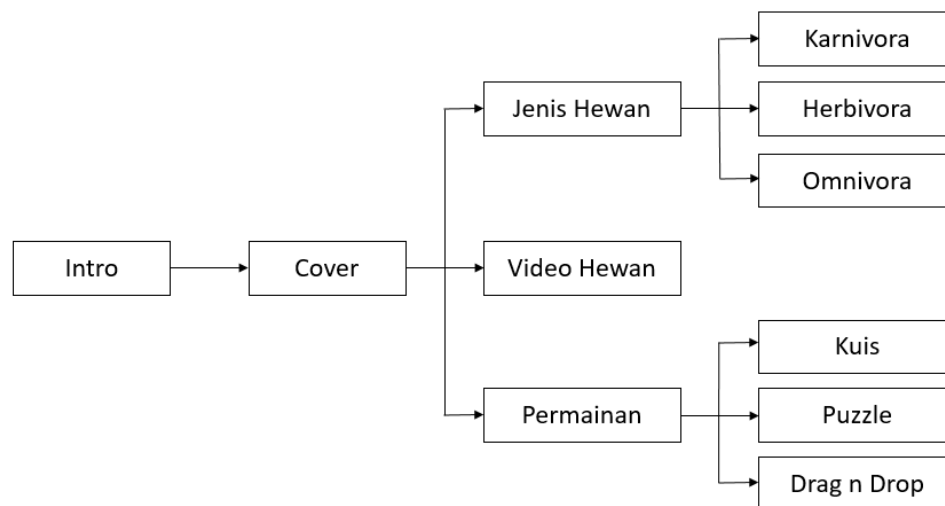
Visual Development Phase

Menyusun Halaman Multimedia Interaktif

5.1 Format Proyek (Susunan Layer)

Setelah memahami area kerja Adobe Animate, pada tahapan selanjutnya dilakukan penyusunan aset visual yang telah dibuat pada *Timeline*. Pada dasarnya terdapat beberapa teknik dalam penyusunan aset visual pada *Timeline* seperti penyusunan menggunakan kode, penyusunan secara *drag and drop* pada *Stage*, menyusun halaman dalam format *Scene* dan menyusun halaman pada *file-file* terpisah. Masing-masing teknik memiliki keuntungan dan kerumitan tersendiri, sebagai contoh menyusun halaman menggunakan kode memiliki fleksibilitas yang tinggi karena dapat diatur secara dinamis sesuai dengan ukuran layar (responsive), namun memiliki tingkat kerumitan pemrograman. Pada buku ini digunakan teknik *drag and drop* dalam penyusunan halaman multimedia interaktif untuk mempermudah pembaca dalam memahami konsep tata letak halaman multimedia interaktif.

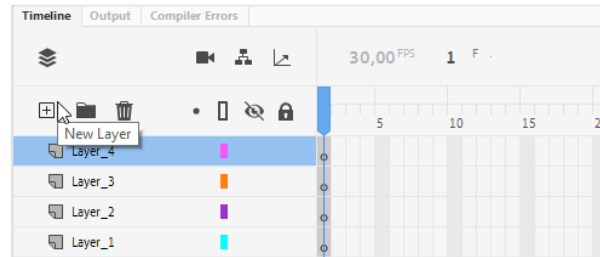
Dalam pengembangan multimedia interaktif, perencanaan awal aplikasi dapat digambarkan dalam sebuah skema navigasi sederhana untuk mempermudah pengembang dalam merencanakan posisi objek dalam *Timeline*. Sebagai contoh pada tutorial berikutnya akan dikembangkan multimedia interaktif berjudul “Tebak Satwa” yang memiliki struktur navigasi halaman sebagai berikut :



Gambar 123. struktur navigasi halaman multimedia interaktif “Tebak Satwa”

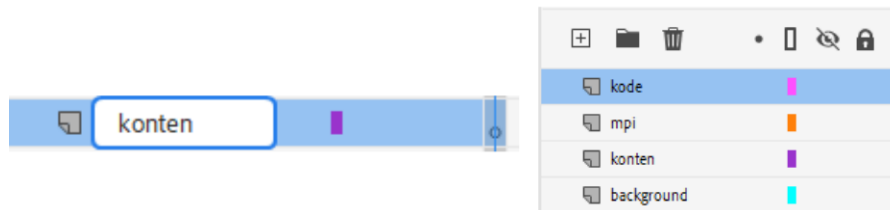
Pada struktur di atas, terdapat beberapa halaman yang harus disusun untuk membentuk aplikasi secara utuh. Pada tahapan selanjutnya diperlukan pengaturan proyek pada Adobe Animate untuk mempermudah penyusunan masing, masing halaman. Perhatikan contoh berikut:

1. Buatlah sebuah *file* baru (**tutorial 8 – Tebak Satwa**)
2. Atur dokumen dengan ukuran layar 1280 x 720 *pixel* dan 30 fps.
3. Buatlah 4 buah *Layer* baru dengan cara menekan tombol + pada panel *Timeline*.



Gambar 124. membuat 4 buah *Layer*

4. Double klik nama *Layer*, dan ubah masing-masing nama *Layer* menjadi *Layer* "**background**", "**konten**", "**mpi**", dan *Layer* "**kode**".



Gambar 125. mengubah nama *Layer*

5. Simpan *file* dan proyek siap untuk ditambahkan aset visual.

Dalam proyek multimedia yang akan dibuat, digunakan 4 *Layer* untuk mempermudah pengaturan aset visual, yaitu :

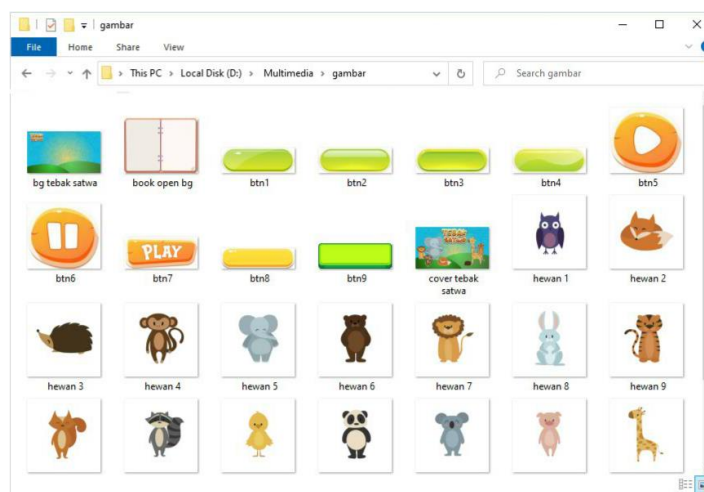
- a. *Layer background*, digunakan untuk meletakkan aset visual berupa gambar latar. Sifatnya dapat berupa gambar statis maupun *MovieClip*.
- b. *Layer konten*, digunakan untuk meletakkan aset visual berupa komponen navigasi seperti tombol, komponen informasi seperti teks, panel informasi, ilustrasi, animasi dan sebagainya.
- c. *Layer mpi*, digunakan untuk meletakkan komponen multimedia pembelajaran interaktif, yang akan dijelaskan pada bab selanjutnya.
- d. *Layer kode*, digunakan untuk menuliskan kode.

Penambahan *Layer* dapat dilakukan secara fleksibel mengikuti kompleksitas halaman yang akan dibuat.

5.2 Penyusunan Halaman

Tahapan selanjutnya adalah menyusun halaman multimedia interaktif, namun sebelum memulai proses penyusunan terlebih dahulu dilakukan persiapan proses impor aset visual yang akan dipakai. Dalam contoh ini digunakan beberapa grafis untuk latar belakang (*background*), tombol, dan beberapa grafis lainnya. Beberapa catatan terkait dengan aset visual yang akan digunakan agar multimedia interaktif berjalan dengan kualitas yang baik, adalah sebagai berikut :

1. Gunakan gambar dengan ukuran yang tepat. Sebagai contoh, untuk gambar latar gunakan gambar berukuran sesuai dengan ukuran proyek, dalam contoh ini menggunakan ukuran 1280x720 *pixel*. Jangan gunakan gambar yang berukuran terlalu besar, atau terlalu kecil. Gambar berukuran besar akan membutuhkan lebih banyak *memory* dan memungkinkan aplikasi berjalan lambat atau gambar tidak muncul di layar, sementara gambar yang terlalu kecil apabila diperbesar akan menghasilkan gambar yang kabur.
2. Gunakan *file* bertipe JPG (resolusi 100%) untuk gambar tanpa transparansi, dan *file* bertipe PNG untuk gambar yang memiliki transparansi.
3. Gunakan gambar yang dibuat secara eksklusif oleh tim pengembang, dalam artian gambar dibuat sendiri. Apabila terpaksa menggunakan gambar yang tersedia secara online, pastikan penggunaan gambar tidak melanggar hak cipta dan gambar tidak memiliki *watermark*.
4. Berikan penamaan *file* yang jelas, sebagai contoh untuk *file* judul digunakan nama *file* “cover tebak satwa.jpg”. Penamaan yang baik akan mempermudah dalam proses pengembangan aplikasi, terlebih apabila aset yang digunakan jumlahnya relatif banyak.
5. Letakkan seluruh gambar yang digunakan dalam *folder* khusus untuk mempermudah proses import.



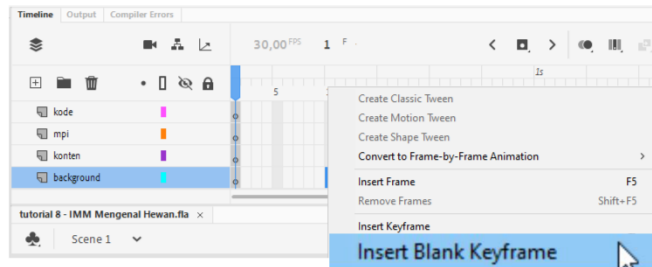
Gambar 126. *folder* gambar dan aset visual yang akan digunakan dalam proyek multimedia

Untuk mulai menyusun halaman multimedia interaktif, perhatikan langkah-langkah berikut :

1. Buka kembali *file (tutorial 8 – Tebak Satwa)* yang telah dibuat sebelumnya.
Pada tahapan awal akan diatur halaman *cover* (judul) aplikasi, sementara halaman intro akan dilewati terlebih dahulu. Pada contoh tutorial ini akan digunakan *Frame* dengan kelipatan 10 untuk mempermudah navigasi aplikasi, yang dimaksud dengan pernyataan ini aplikasi akan dimulai dari *Frame 10* (untuk halaman judul), *Frame 20* (halaman berikutnya) dan seterusnya.

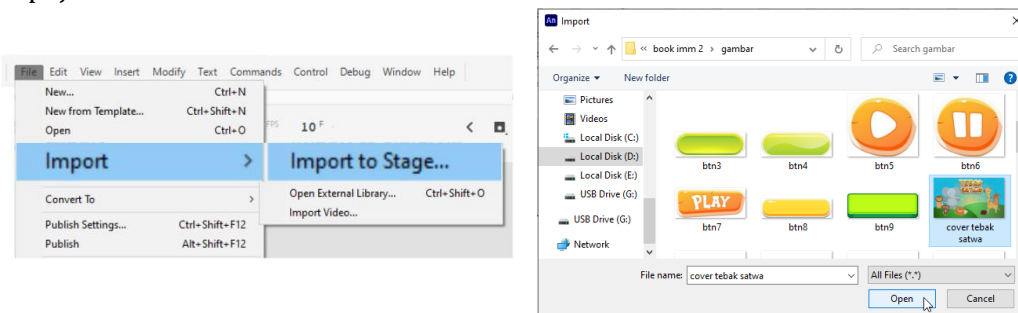
Beberapa literatur pengembangan aplikasi dengan Adobe Animate / Adobe Flash menggunakan teknik *Scene*. Masing-masing halaman diletakkan pada *Scene* yang berbeda. Teknik ini juga dapat dipakai, namun berdasarkan pengalaman penulis penggunaan *Scene* tidak direkomendasikan karena setiap perpindahan *Scene* aplikasi akan mereset variabel-variabel tertentu. Dalam buku ini teknik penyusunan halaman adalah dalam satu *Scene* yang sama, halaman akan disusun pada *Frame* dengan kelipatan tertentu.

2. Untuk mulai menambahkan latar belakang untuk halaman judul, klik kanan pada **Frame 10 Layer background** kemudian pilih *Insert Blank Keyframe*.



Gambar 127. menambahkan *Blank Keyframe* pada *Frame 10 Layer background*

3. Untuk menambahkan latar, klik menu **File > Import > Import to Stage**. Pilih *file* latar yang telah disiapkan sebelumnya, kemudian tekan **Open**. Setelah latar halaman judul selesai ditambahkan ke layar, atur posisinya tepat pada kordinat x:0 dan y:0 atau tepat di pojok kiri atas.



Gambar 128. mengimpor latar halaman judul

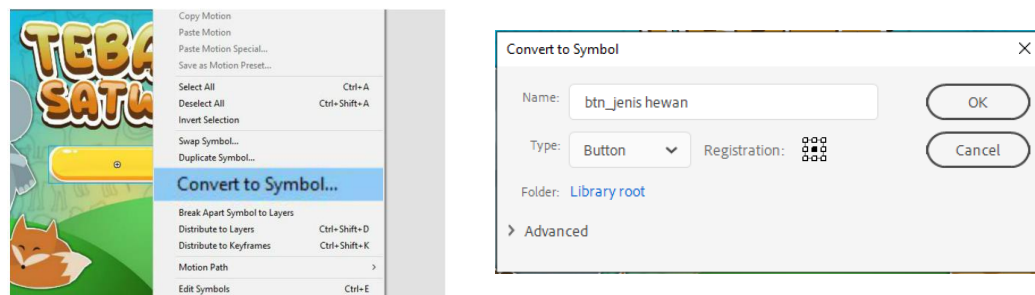
4. Sesuai dengan struktur navigasi halaman yang direncanakan, maka pada halaman judul akan terdapat 3 tombol navigasi utama. Oleh karena itu, akan ditambahkan 3 buah tombol. Untuk mendapatkan prinsip kesatuan (*unity*) yang baik, cukup membuat satu buah tombol terlebih dahulu, yang selanjutnya dapat diduplikasi.

Untuk membuat tombol, klik **Frame 10 Layer konten** tambahkan **Blank Keyframe** (lihat langkah 3), kemudian **import** gambar latar tombol ke *Stage* (lihat langkah 4) lalu letakkan pada tengah layar. Pada contoh di buku ini, digunakan gambar tombol tanpa teks, karena teks akan ditambahkan secara manual dengan menggunakan *text tool*.



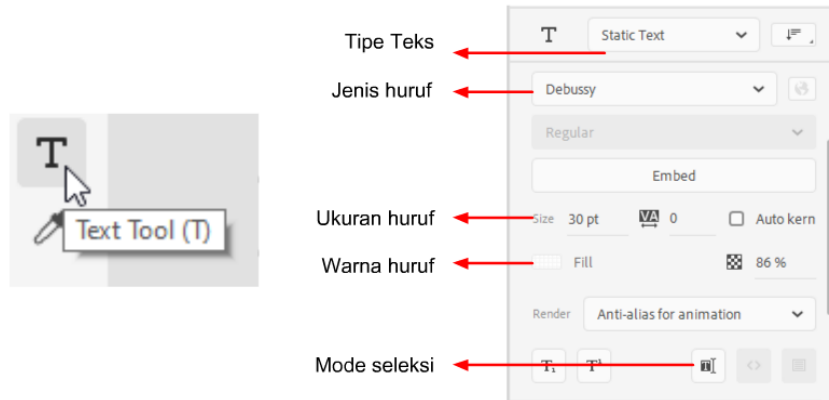
Gambar 129. peletakan latar tombol pada *Stage*

5. Untuk mengubah gambar tersebut menjadi tombol, klik kanan gambar latar tombol kemudian pilih opsi **Convert to Symbol**. Ketikkan nama "btn_jenis hewan", type "Button" dan registrasi di tengah, lalu tekan **OK**.



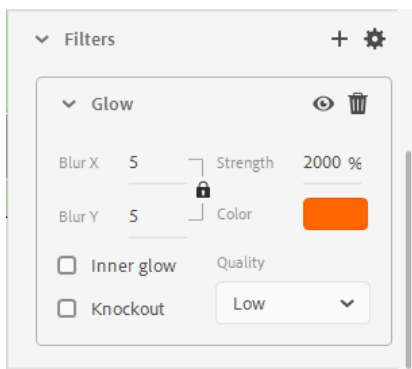
Gambar 130. mengubah gambar menjadi tombol

6. Edit tombol tersebut dengan melakukan double klik pada tombol. Selanjutnya untuk menambahkan teks (label) pada tombol, klik **Text Tool**. Pada panel **Properties** pastikan tipe teks terpilih opsi **Static Text**. Atur jenis, ukuran dan warna huruf, pastikan label dapat dibaca dengan mudah oleh pengguna. Pastikan opsi **selectable** (mode seleksi) dalam keadaan mati, apabila opsi tersebut aktif maka saat *mouse* melintas di atas teks, kursor akan berubah menjadi mode seleksi sehingga menghasilkan pengalaman pengguna yang kurang baik.



Gambar 131. mengatur jenis teks

Untuk meningkatkan keterbacaan, juga dapat ditambahkan efek *outline* dengan mengatur **filter**. Klik tombol + kemudian tambahkan **filter Glow**. Atur **Blur** menjadi 5 dan **Strength** 2000% untuk menghasilkan *outline* yang tajam. Pilih warna yang lebih gelap dari warna latar untuk meningkatkan keterbacaan.



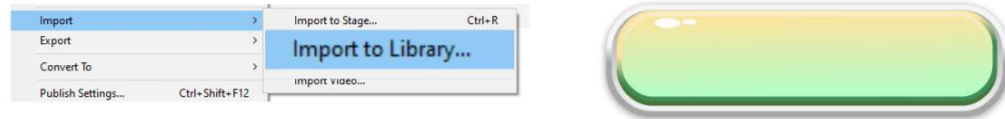
Gambar 132. mengatur efek *outline* teks

Klik pada *Stage* dan ketikkan “Jenis Hewan”. Atur kembali posisi teks dan untuk mempermudah proses selanjutnya, pastikan teks berada tepat di tengah tombol dan menggunakan **align center**.



Gambar 133. tombol “Jenis Hewan”

7. Pada tahapan ini, dapat juga ditambahkan efek tombol ketika dalam kondisi **over** dan **down**. Untuk melakukannya impor latar untuk tombol baru ke *Library* (dalam contoh di buku ini digunakan latar tombol yang lebih terang ketika kondisi *over*) (pilih menu **File > Import > Import to Library**).



Gambar 134. mengimpor gambar ke *Library*

Klik kanan **Frame over** dan tambahkan **Keyframe**, sehingga pada saat ini *Frame up* dan *over* memiliki konten yang sama. Selanjutnya pada *Frame over* **klik kanan background tombol**, kemudian pilih opsi **Swap Bitmap**. Pilih gambar latar tombol yang baru dan klik **OK**.



Gambar 135. mengubah *bitmap* latar tombol

Klik teks “Jenis Hewan”, kemudian atur kembali warna dan efek *glow* untuk menghasilkan keterbacaan yang baik. Dengan cara yang sama, impor latar baru (latar tombol yang memiliki efek ditekan/masuk sedikit ke dalam) dan tambahkan **Keyframe** pada *Frame down*, lalu ubah latar tombol untuk menghasilkan efek tombol ditekan.



Gambar 136. mengatur tombol “Jenis Hewan”

Langkah no 4 sampai 7 merupakan langkah standar membuat tombol bertipe *Call to Action Button (CTA button)*. Pada tahapan berikutnya di buku ini akan ditulis secara ringkas menjadi “buatlah sebuah tombol dengan label “Jenis Hewan””.

8. Untuk membuat tombol lainnya yang sejenis, tidak perlu dibuat dari awal, akan tetapi dapat dilakukan proses duplikasi simbol. Pada struktur navigasi halaman judul direncanakan terdapat 3 tombol navigasi utama. Satu tombol navigasi yaitu tombol “Jenis Hewan” sudah dibuat, sehingga dapat dijadikan sebagai acuan untuk membuat 2 tombol lainnya.

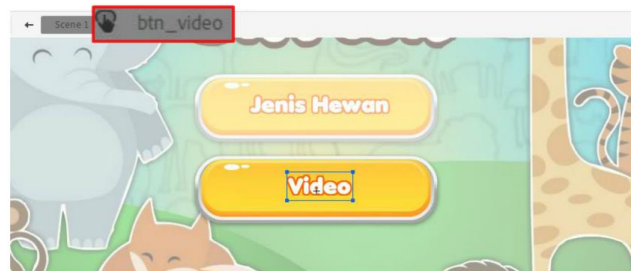
Proses duplikasi simbol dimulai dengan teknik *copy-paste*. Klik tombol “Jenis Hewan” lalu tekan **Ctrl+C** untuk mengcopy dan tekan **Ctrl+P** untuk mempaste. Atur posisi tombol baru tepat di bawah tombol “Jenis Hewan”. Untuk menduplikasi tombol, **klik**

kanan tombol baru kemudian pilih **Duplicate Symbol**. Ubah nama simbol menjadi “btn_video”, lalu tekan **OK**.



Gambar 137. menduplikasi simbol

Setelah tombol terduplikasi maka tombol dapat diedit dengan cara double klik tombol. **Double klik** tombol “btn_video” untuk memasuki mode edit simbol. Selanjutnya **double klik** teks “Jenis Hewan” dan edit menjadi “Video”. Klik **Frame over** ubah juga teks yang ada, demikian halnya dengan teks pada **Frame down**. Setelah semua teks selesai diubah, maka akan didapatkan tombol baru yang memiliki format yang sama dengan tombol “Jenis Hewan”. Tekan tombol **Scene 1** atau **Ctrl+E** untuk keluar dari mode edit simbol.



Gambar 138. mengedit tombol “btn_video”

Proses duplikasi tersebut ditujukan untuk membentuk prinsip *unity* (kesatuan) pada tata letak halaman judul. Dengan duplikasi visualisasi tombol dari sisi latar belakang dan label akan seragam dan menunjukkan hirarki yang setara.

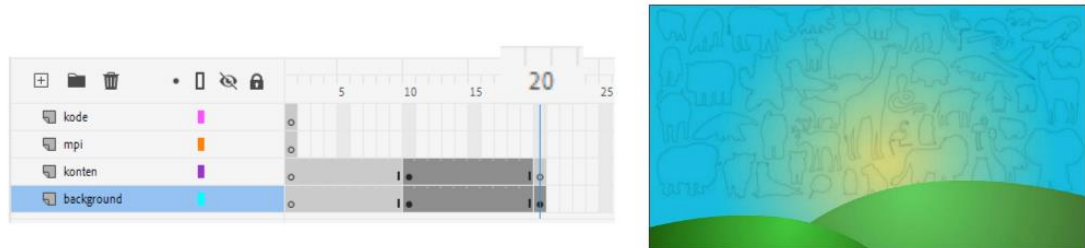
9. Dengan cara yang sama dengan langkah 8, buatlah duplikasi tombol “btn_permainan” dan letakkan di bawah tombol “btn_video”.



Gambar 139. tombol navigasi pada halaman judul.

10. Tahapan berikutnya adalah menambahkan halaman kedua yaitu halaman “jenis hewan”. Seperti pada penjelasan di langkah 1, pengembangan aplikasi multimedia interaktif pada contoh ini menggunakan *Frame* kelipatan 10, sehingga halaman jenis hewan akan diletakkan di *Frame* 20.

Seleksi **Frame 20 Layer background** kemudian tambahkan **Blank Keyframe**. Impor gambar untuk latar belakang halaman tersebut.



Gambar 140. menambahkan halaman “jenis satwa”

Seleksi **Frame 20 Layer konten** kemudian tambahkan **Blank Keyframe**. Pada *Layer* konten tambahkan 3 buah tombol, yaitu tombol “btn_karnivora”, “btn_omnivora” dan “btn_herbivora”. Pada halaman ini juga dapat ditambahkan **Static Text** “jenis Satwa” dengan jenis huruf dan ukuran yang telah ditentukan sebelumnya.



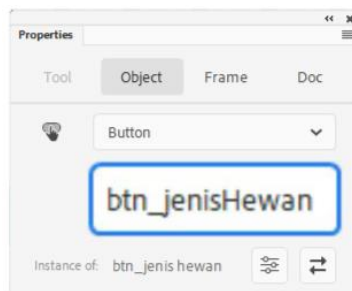
Gambar 141. menambahkan konten halaman jenis satwa

Proses penambahan halaman selanjutnya akan dijelaskan pada bab-bab selanjutnya dengan teknik dan metode yang relatif sama.

5.3 Penambahan *Instance name*

Seperti pada penjelasan bab sebelumnya terkait *instance name*, bahwa setiap objek perlu diberikan identifikasi agar dapat diakses oleh kode. Sebelum memasuki tahapan pemrograman, maka masing-masing objek yang menjadi objek interaktif wajib diidentifikasi dengan penambahan *instance name*. Untuk menambahkan *instance name* perhatikan langkah-langkah berikut :

1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**) yang telah memiliki 2 buah halaman, yaitu halaman judul dan halaman “Jenis Hewan”. Penambahan *instance name* dapat dilakukan secara bertahap, satu halaman demi satu halaman.
2. Klik **Frame 10** untuk berpindah ke halaman judul. Klik tombol **btn_jenis hewan** kemudian buka panel **Properties**. Pada kolom *instance name* ketikkan “btn_jenisHewan”.



Gambar 142. penambahan *instance name*

3. Dengan cara yang sama, tambahkan *instance name* “btn_video” dan “btn_permainan”.
4. Klik **Frame 20** kemudian tambahkan *instance name* “btn_karnivora”, “btn_omnivora” dan “btn_herbivora”.



Gambar 143. penambahan *instance name*

Dengan penambahan *instance name* pada masing-masing tombol maka tahapan dapat dilanjutkan ke *programming phase*.

BAB 6

Programming Phase

Pengembangan Multimedia Interaktif dengan *MPI Component*

6.1 Konsep dasar *MPI Component*

Proses pemrograman dalam pengembangan aplikasi melibatkan pemahaman terkait bahasa pemrograman tertentu. Demikian halnya dengan pengembangan aplikasi multimedia interaktif menggunakan Adobe Animate yang secara spesifik menggunakan bahasa pemrograman *Actionscript*. *Actionscript* adalah bahasa pemrograman berorientasi objek yang awalnya dikembangkan oleh Macromedia dan kemudian diakuisisi oleh Adobe. *Actionscript* mengadaptasi kode ECMAScript (yang merupakan superset dari sintaks dan semantik dari bahasa yang lebih dikenal sebagai JavaScript). *Actionscript* digunakan terutama untuk pengembangan situs *web* dan perangkat lunak yang menargetkan *platform* Adobe Flash, awalnya digunakan pada halaman *web* dalam bentuk *file* SWF yang disematkan dan pada periode berikutnya dapat dijalankan di berbagai *platform* seperti PC, Android dan IOS.

Bagi pemula, memahami sebuah kode membutuhkan periode waktu yang panjang karena diperlukan pemahaman terhadap sintaks dan logika pemrograman. Sebagai contoh untuk mengaktifkan sebuah tombol navigasi yang ketika ditekan akan membawa ke halaman lain maka diperlukan beberapa baris kode sebagai berikut :

```
tombol.addEventListener(MouseEvent.CLICK, pindahHalaman);

function pindahHalaman(e:MouseEvent):void{
    tombol.removeEventListener(MouseEvent.CLICK, pindahHalaman);
    gotoAndStop(20);
}
```

Kode di atas bisa dikatakan cukup kompleks hanya untuk menjalankan sebuah perintah tombol yang ditekan oleh pengguna. Ketika melibatkan banyak tombol yang ada di sebuah halaman, akan berimplikasi pada jumlah baris kode yang digunakan. Oleh karena alasan tersebut, penulis menyusun sebuah kode yang terkompilasi untuk mempermudah pengembang aplikasi (khususnya pemula) dalam proses pemrograman. Kode terkompilasi tersebut dinamakan *MPI Component* (Komponen Multimedia Pembelajaran Interaktif). Dengan penggunaan *MPI Component*, kode di atas dapat dituliskan cukup 1 baris saja, yaitu :

```
mpi.click("tombol", 20);
```

MPI Component berupa sebuah *compiled clip* yang dapat diletakkan di *Timeline* proyek aplikasi multimedia yang dikembangkan. Di dalamnya terdapat berbagai jenis kode khusus untuk pengembangan aplikasi multimedia interaktif, kode-kode yang telah diringkas, termasuk manajemen *memory* aplikasi yang baik, sehingga pengguna pemula tidak harus memikirkan pengaturan *memory* aplikasi (khususnya penggunaan *listener*).

Untuk menggunakan *MPI Component*, pengguna harus mendownload file bertipe FLA dari situs <https://www.wandah.org/download/code/MPI-component fla>. File FLA tersebut dapat dibuka dengan aplikasi Adobe Animate versi 2017 ke atas. Setelah file terbuka, di dalamnya terdapat beberapa contoh penggunaan *MPI Component* dan beberapa aset yang dapat digunakan secara bebas oleh siapapun.



Gambar 144. simbol *MPI Component* dalam format *precompiled clip*

6.2 Menambahkan *MPI Component* ke file proyek multimedia interaktif

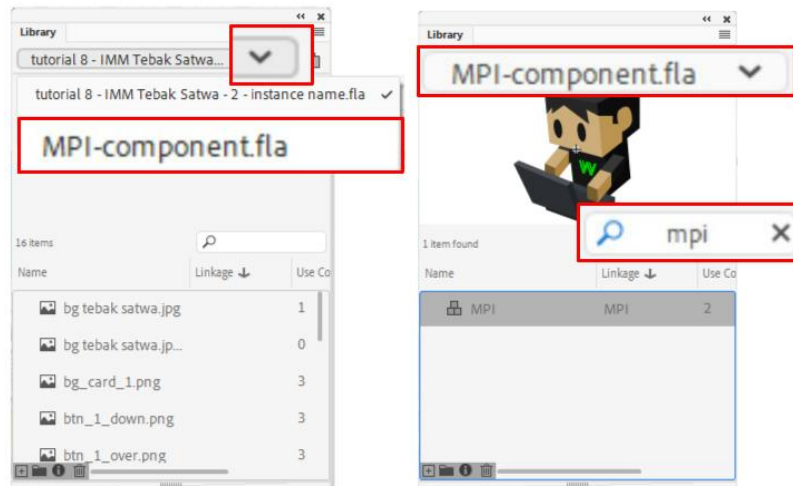
Pada tahapan berikut, akan dijelaskan cara menambahkan *MPI Component* ke file proyek multimedia interaktif. Perhatikan langkah-langkah berikut :

1. Download file **MPI-component fla** dari situs [wandah.org](https://www.wandah.org), kemudian buka file tersebut.



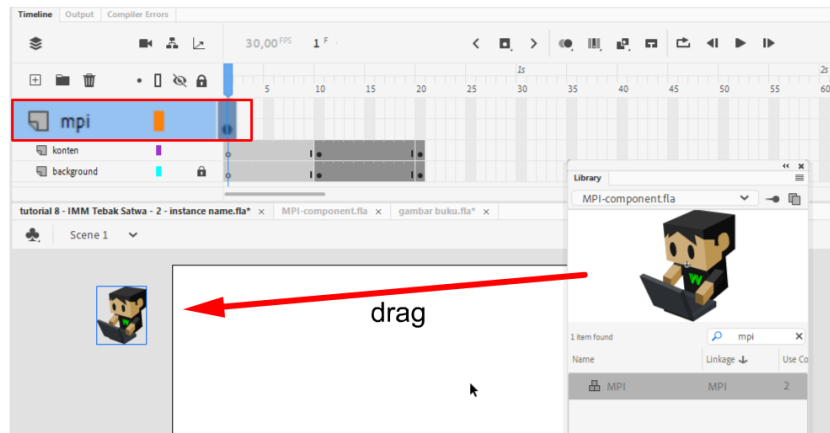
Gambar 145. halaman *download MPI Component*

2. Buka kembali file (**tutorial 8 – Tebak Satwa**) yang telah memiliki 2 buah halaman, yaitu halaman judul dan halaman “Jenis Hewan” serta telah memiliki **instance name**.
3. Untuk menggunakan *MPI Component* diperlukan *precompiled clip* yang berasal dari *Library file* MPI-Component fla. Oleh karena itu buka panel **Library (Ctrl+L)** kemudian pilih *Library file* MPI-Component.



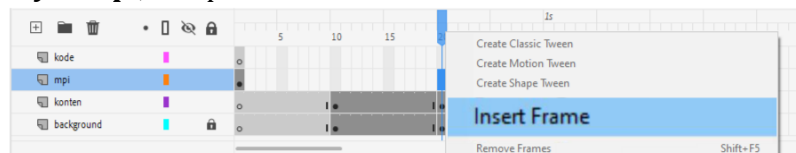
Gambar 146. membuka *Library file* MPI-component fla

Klik **Frame 1 Layer mpi**, kemudian **drag** simbol MPI dari *Library* ke *Stage*. Peletakan simbol mpi relatif bebas, simbol dapat diletakkan di luar *Stage* atau di area *Stage*.



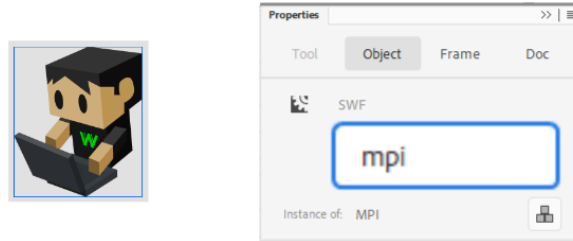
Gambar 147. menambahkan *MPI Component*

4. Simbol MPI harus berada pada seluruh *Frame* di *Timeline*. Oleh karena itu, **klik kanan** **Frame 20 Layer mpi**, dan pilih **Insert Frame**.



Gambar 148. menambahkan *Frame*

5. Selanjutnya agar simbol MPI aktif, klik simbol dan **tambahkan instance name** "mpi". Penambahan *instance name* merupakan tahapan yang sangat penting agar simbol dapat dikenali oleh kode *Actionscript*.



Gambar 149. Penambahan *instance name* pada simbol mpi

6.3 Navigasi Halaman

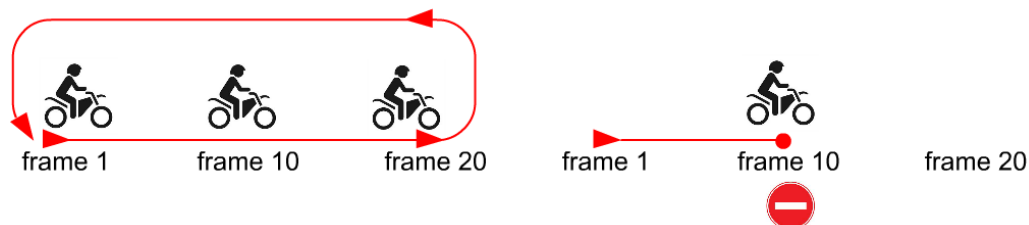
Setelah *MPI Component* telah ditambahkan ke proyek multimedia interaktif yang sedang dikembangkan, maka kode-kode di dalamnya dapat diakses dengan mudah, termasuk kode untuk navigasi halaman. Kode untuk mengaktivasi tombol navigasi dalam *MPI Component* adalah :

```
mpi.click("instance name tombol", frame tujuan);
```

Perhatikan implementasi kode di atas pada proyek multimedia “Tebak Satwa” berikut ini :

1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**) yang telah memiliki 2 buah halaman, dan *MPI Component*.
2. Jalankan aplikasi dengan menekan tombol **Ctrl+Enter**, maka akan dihasilkan aplikasi yang berulang (*looping*). Aplikasi yang diharapkan, seharusnya berhenti pada halaman judul (*Frame 10*). Oleh karena itu perlu ditambahkan sebuah kode agar aplikasi dapat berhenti.

Kode (*Actionscript*) dapat diibaratkan sebagai rambu lalu lintas yang mengatur jalannya sebuah kendaraan. Bayangkan aplikasi yang sedang dibuat adalah sebuah sepeda motor, yang bergerak di sirkuit yang melingkar mulai dari *Frame 1* menuju *Frame 20* dan kembali lagi ke *Frame 1*. Sepeda motor akan terus berjalan melintasi sirkuit dan berputar secara terus menerus. Selanjutnya pada *Frame 10* akan ditambahkan rambu lalulintas agar sepeda motor berhenti, maka rambu lalulintas tersebut diletakkan tepat di *Frame 10*. Dengan penambahan rambu tersebut sepeda motor akan bergerak dari *Frame 1* dan berhenti tepat di *Frame 10*.



Gambar 150. representasi sebelum penambahan kode (kiri) dan setelah penambahan kode (kanan)

Untuk menghentikan aplikasi di halaman judul, **klik kanan *Frame 10 Layer* kode** kemudian tambahkan ***Blank Keyframe***. Klik ***Frame 10 Layer* kode**, kemudian buka panel ***Action (F9)*** dan ketikkan kode berikut :

```
stop();
```

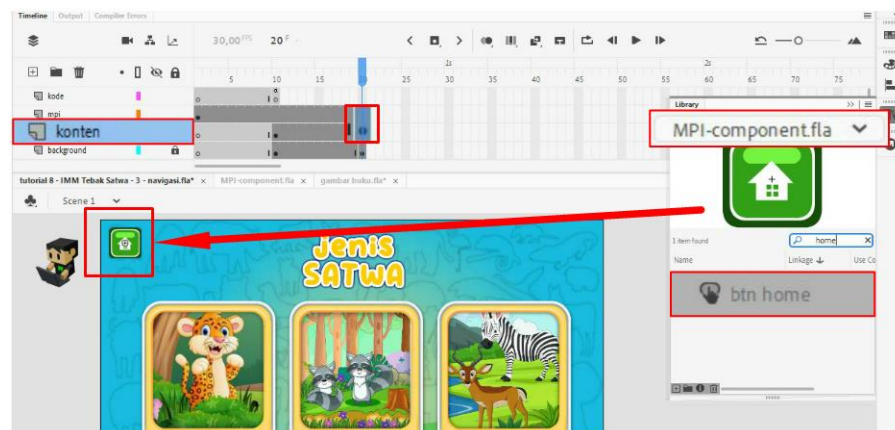
3. Dengan penambahan kode `stop();` tersebut ketika aplikasi dijalankan (dengan menekan **Ctrl+Enter**) maka aplikasi akan berhenti di halaman judul. Selanjutnya ditambahkan kode untuk 3 tombol navigasi yang terdapat di halaman judul. Pada contoh ini, baru ditambahkan satu kode terlebih dahulu, hal ini dikarenakan halaman yang tersusun baru 2 yaitu halaman judul dan halaman jenis hewan, sedangkan 2 halaman lainnya belum dibuat.

Untuk menambahkan kode navigasi pada tombol “jenis hewan”, **klik kembali *Frame 10 Layer* kode**, kemudian buka panel ***Action*** dan tambahkan kode berikut :

```
mpi.click("btn_jenisHewan", 20);
```

Jalankan kembali aplikasi dengan menekan **Ctrl+Enter**, maka apabila tombol “Jenis Hewan” di klik, aplikasi akan segera berpindah ke *Frame 20* (tempat halaman Jenis Hewan). Pada saat aplikasi berada di halaman jenis hewan, pengguna belum dapat melakukan apapun karena tidak ada kode pada halaman tersebut. Oleh karena itu diperlukan sebuah tombol untuk kembali ke halaman judul.

4. Untuk menambahkan tombol kembali (seringkali diwujudkan dalam bentuk tombol “home”), **klik *Frame 20 Layer* konten**, kemudian buatlah sebuah tombol “home”, letakkan di pojok kiri atas. Atau cara cepat adalah dengan membuka ***Library file MPI-component*** dan mencari tombol *home*, lalu *mendrag* tombol tersebut ke *Stage*.



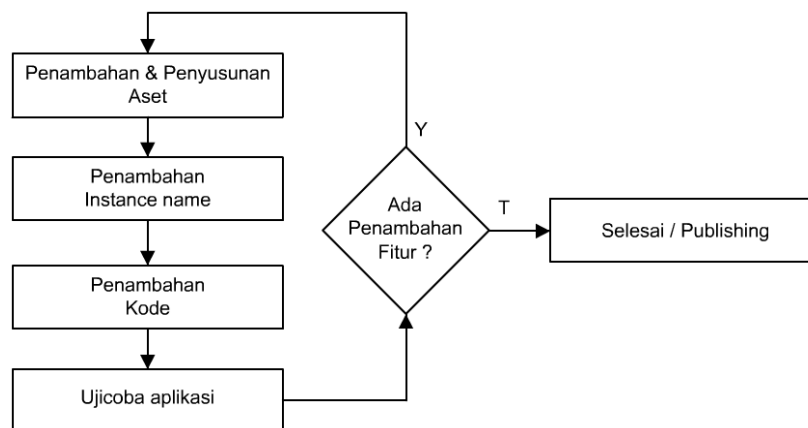
Gambar 151. menambahkan tombol home dari *Library MPI-Component*

5. Klik tombol *home* tersebut, kemudian **tambahkan instance name** “btn_home”. Tambahkan juga *instance name* untuk 3 tombol hewan, yaitu “btn_karnivora”, “btn_omnivora” dan “btn_herbivora” (apabila belum dilakukan).
6. Untuk mengaktifkan tombol “home” perlu ditambahkan sebuah kode. **Klik Frame 20 Layer kode** tambahkan **Blank Keyframe**. Selanjutnya buka panel **Action** dan ketikkan kode berikut :

```
mpi.click("btn_home", 10);
```

Jalankan kembali aplikasi, dan saat ini aplikasi sudah memiliki sistem navigasi dasar. Pengguna dapat mengakses tombol “Jenis Hewan” untuk menuju ke halaman jenis hewan dan kembali ke halaman judul dengan menekan tombol *home*.

Sampai dengan langkah navigasi dasar tersebut dapat disimpulkan bahwa proses pengembangan aplikasi multimedia interaktif merupakan sebuah proses perulangan yang secara ringkas dapat digambarkan dengan *flowchart* sebagai berikut :



Gambar 152. *flowchart* pengembangan multimedia interaktif

6.4 Penambahan Halaman

Pada contoh berikut akan dilakukan penambahan halaman “hewan karnivora” untuk memperkuat konsep dasar pengembangan multimedia interaktif. Pada halaman “hewan karnivora” akan ditambahkan sebuah konten statik berupa gambar, teks dan tombol. Sedangkan contoh penambahan konten yang lebih interaktif akan dibahas pada sub bab berikutnya. Perhatikan langkah-langkah berikut untuk penambahan sebuah halaman baru:

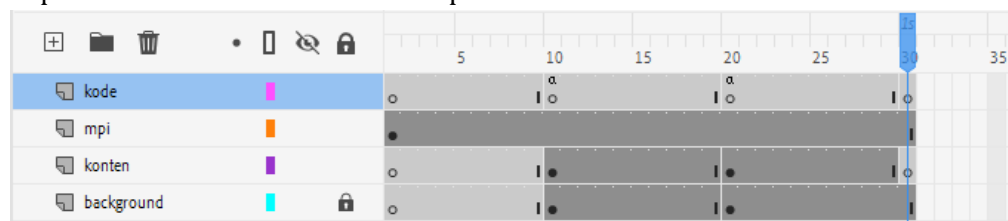
1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**).
2. Penambahan halaman “hewan karnivora” akan ditambahkan pada *Frame 30*, oleh karena itu klik **Frame 30 Layer background**, dan tambahkan **Frame** (*Insert Frame*). Hal ini akan menyebabkan *background* sebelumnya muncul di *Frame 30*. Apabila ingin

mengubah *background*, maka tahapan yang dilakukan adalah menambahkan **Blank Keyframe** dan mengimpor *background* baru.

Klik **Frame 30 Layer konten** dan tambahkan **Blank Keyframe**, sehingga konten pada *Frame* ini kosong, dan siap untuk ditambahkan konten baru.

Klik **Frame 30 Layer mpi** dan tambahkan **Frame (Insert Frame)**. Untuk *Layer mpi* tidak perlu adanya penambahan *Keyframe*. *Layer mpi* cukup memiliki 1 buah *Keyframe* (pada *Frame* 1) dan berisi hanya simbol “mpi”.

Klik **Frame 30 Layer kode** dan tambahkan **Blank Keyframe**. Setiap halaman membutuhkan kode, dan kode diletakkan pada sebuah **Blank Keyframe** untuk mempermudah memahami struktur aplikasi.



Gambar 153. struktur *Timeline* untuk penambahan halaman baru

3. Klik **Frame 30 Layer konten**, selanjutnya tambahkan elemen informasi terkait halaman. Pada halaman ini dapat ditambahkan gambar statik (dengan cara mengimpor ke *Stage*), informasi terkait konten dalam format teks (*Static Text*), tombol “Contoh Karnivora” dan tombol “kembali”, atau dapat juga ditambahkan animasi dalam format *MovieClip*.



Gambar 154. konten *Frame* 30

4. **Tambahkan *instance name*** pada tombol yang ada. Pada contoh di atas ditambahkan *instance name* “btn_kembali” dan “btn_contohKarnivora”.

5. Klik **Frame 20 Layer kode**, kemudian tambahkan kode untuk tombol “btn_karnivora” sebagai berikut :

```
mpi.click("btn_karnivora", 30);
```

Selanjutnya klik **Frame 30 Layer kode**, dan tambahkan kode untuk tombol “btn_kembali” agar ketika diklik dapat kembali ke halaman jenis hewan (*Frame 20*).

```
mpi.click("btn_kembali", 20);
```

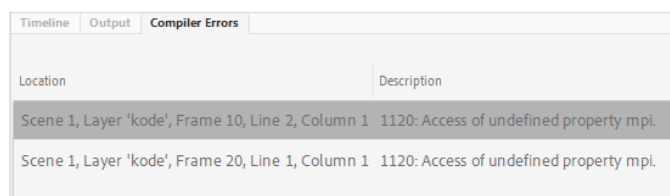
6. Jalankan aplikasi dengan menekan **Ctrl+Enter**, maka akan halaman karnivora dapat diakses dengan menekan tombol “jenis hewan” pada halaman judul, lalu menekan tombol “karnivora”.

Pada dasarnya penambahan halaman lainnya seperti halaman “hewan omnivora” dan “hewan herbivora” memiliki langkah-langkah yang sama, sehingga tidak dijelaskan secara mendetail pada buku ini. Halaman “hewan omnivora” diletakkan pada *Frame 40* dan halaman “hewan herbivora” diletakkan pada *Frame 50*. Sedangkan halaman “contoh karnivora” akan dibahas pada sub bab berikutnya.



Gambar 155. halaman “hewan omnivora” (*Frame 40*)
dan halaman “hewan herbivora” (*Frame 50*)

Dalam menjalankan aplikasi seringkali terjadi kesalahan-kesalahan dasar. Kesalahan yang paling umum adalah belum menambahkan *instance name* atau kesalahan ketik. Segala kesalahan akan dimunculkan pada panel **Compiler Errors** sebagaimana contoh berikut:



Gambar 156. pesan kesalahan yang muncul
akibat penulisan *instance name* yang belum dilakukan

Ketika terjadi kesalahan aplikasi akan berjalan secara terus menerus (*looping*). Pada kondisi ini *developer* pemula tidak perlu panik, cukup membaca kesalahan yang muncul di panel *compiler error*, lalu double klik pesan *error* untuk merujuk baris yang mengalami kesalahan.

6.5 Transisi Halaman

Dalam sebuah aplikasi multimedia interaktif, seringkali ditambahkan efek perpindahan halaman untuk mendapatkan tampilan yang estetik. Transisi dapat berupa efek *fade in-fade out*, efek *dissolve*, efek *masking/wipe*, dan sebagainya. *MPI Component* menyediakan kode singkat untuk menambahkan transisi, yaitu dengan kode :

```
mpi.setTransisi("fade");
```

Sedangkan untuk menonaktifkan transisi atau membuang efek transisi cukup dengan kode :

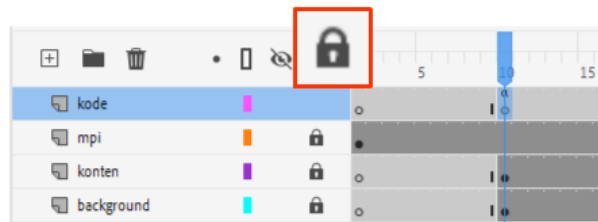
```
mpi.setTransisi("");
```

Kode tersebut dapat ditambahkan pada **Frame 10 Layer kode** untuk mengaktifkan atau menonaktifkan efek transisi. Parameter yang dapat dimasukkan ke dalam fungsi `setTransisi` adalah "fade", "wipe" dan "mc".

6.5.1 Transisi dengan Efek *MovieClip*

Untuk menambahkan efek transisi yang lebih menarik, misalnya menambahkan efek transisi dengan animasi, maka dapat digunakan sebuah *MovieClip*. *MovieClip* yang digunakan memiliki animasi membuka dan menutup, yang akan digunakan sebagai media pergantian halaman. Untuk lebih jelasnya perhatikan langkah-langkah berikut :

1. Buka kembali file (**tutorial 8 – Tebak Satwa**).
2. Klik **Frame 10 Layer kode**. *MovieClip* yang akan dijadikan sebagai efek transisi akan dibuat pada *Layer* teratas, untuk sementara dapat digunakan *Frame 10 Layer Action*. Untuk mempermudah proses, *Layer* lain dapat dikunci (*lock*) terlebih dahulu agar tidak terseleksi.



Gambar 157. mengunci *Layer*

Gambar transisi dapat dibuat secara langsung menggunakan *drawing tool* atau dengan mengimpor gambar. Untuk efek transisi membuka dan menutup, pada buku ini digunakan gambar tirai. **Impor gambar ke Stage**, kemudian letakkan di sebelah kiri luar *Stage*. **Copy-paste** gambar tersebut dan letakkan di sebelah kanan luar *Stage*, dengan demikian akan ada 2 gambar tirai di sebelah kanan dan kiri *Stage*.



Gambar 158. posisi tirai untuk efek transisi *MovieClip*

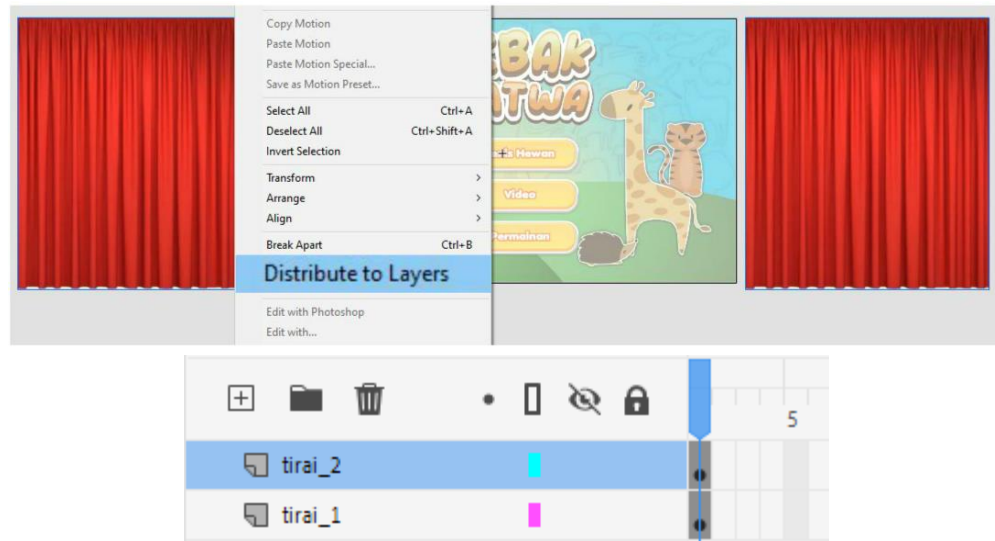
Seleksi kedua gambar tirai tersebut kemudian **Convert to symbol** menjadi **MovieClip** dengan nama “**transisiMC**”. Pastikan titik registrasi dari *MovieClip* adalah tepat di tengah, hal ini penting karena perhitungan lokasi *MovieClip* berpusat pada titik registrasi.

Selanjutnya **dobel klik *MovieClip* transisiMC** untuk mengeditnya. Setelah masuk ke mode edit *symbol*, gambar tirai akan dianimasikan dengan teknik *motion tween*. Syarat pertama dari animasi *motion tween* adalah gambar harus *diconvert* menjadi *symbol*. Oleh karena itu, **klik gambar tirai sebelah kiri**, kemudian **convert** menjadi **MovieClip** “**tirai1**”. Selanjutnya **klik gambar tirai sebelah kanan**, kemudian **convert** menjadi **MovieClip** “**tirai2**”.



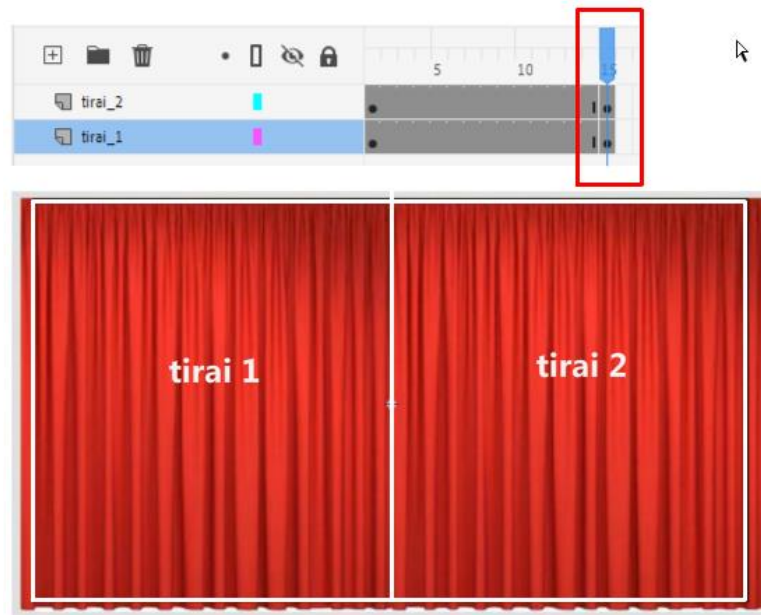
Gambar 159. mengubah gambar menjadi *MovieClip*

Syarat kedua animasi *motion tween* adalah sebuah *Keyframe* hanya boleh ditempati oleh satu simbol. Pada contoh ini satu *Keyframe* masih ditempati oleh dua *MovieClip*, oleh karena itu masing-masing *MovieClip* harus didistribusikan ke *Layer* sendiri-sendiri. Untuk melakukannya dengan mudah, **seleksi kedua *MovieClip***, kemudian **klik kanan** dan **pilih *Distribute to Layers***, maka masing-masing *MovieClip* akan memiliki *Layer* tersendiri



Gambar 160. mendistribusikan *MovieClip* ke masing-masing *Layer*

Syarat ketiga animasi *motion tween* adalah sebuah objek yang bergerak memiliki setidaknya 2 buah *Keyframe*. Efek transisi yang akan dibuat adalah tirai yang menutup kemudian terbuka. Untuk membuat efek gerakan tersebut, **klik kanan *Frame 15* kedua *Layer***, kemudian tambahkan ***Keyframe***. Kemudian **geser** posisi masing-masing ***MovieClip* tirai** menuju ke tengah menutupi *Stage*.



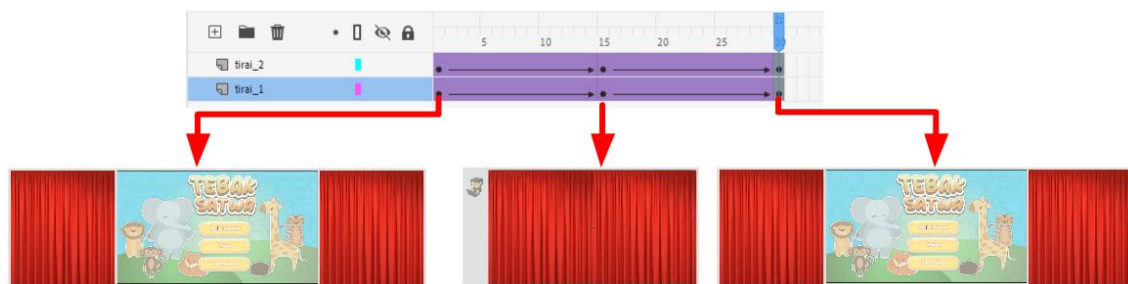
Gambar 161. posisi objek di *Frame 15*

Untuk menjadikannya *motion tween*, **klik kanan pada Frame 10** (*Frame* berapapun di antara *Frame* 1 dan *Frame* 15) **Layer tirai_1**, kemudian pilih **Create Classic Tween**. Demikian halnya dengan *Layer* tirai_2, **klik kanan pada Frame 10 Layer tirai 2** dan pilih **Create Classic Tween**, maka animasi tirai menutup akan terbentuk.



Gambar 162. menambahkan animasi *motion tween*

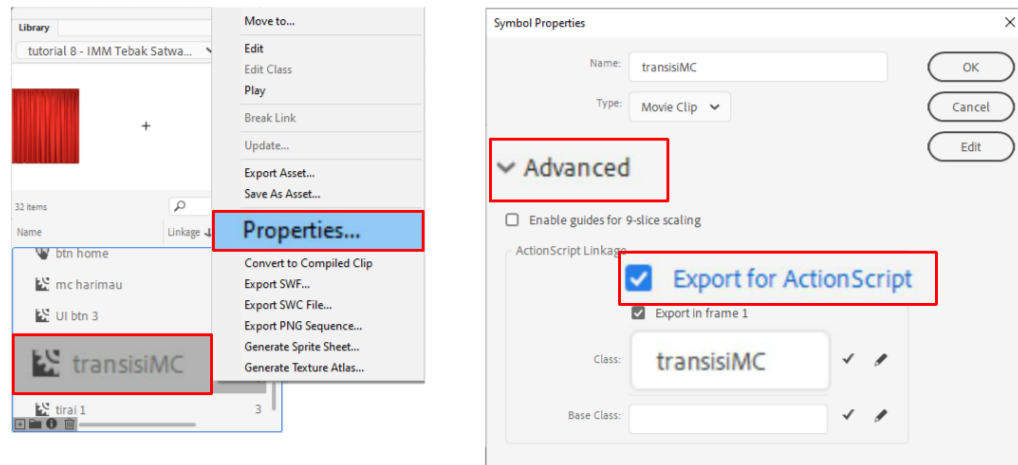
Selanjutnya untuk membuat animasi tirai kembali terbuka, **klik kanan Frame 30 kedua Layer** kemudian tambahkan **Keyframe**. Geser posisi masing-masing tirai kembali ke luar *Stage*. Selanjutnya tambahkan **Classic Tween** agar animasi tirai terbuka terbentuk.



Gambar 163. animasi tirai menutup dan membuka

3. Setelah animasi terbentuk, kembali ke *Stage* utama dengan menekan **Ctrl+E**. Selanjutnya **seleksi MovieClip transisiMC** dan **hapus dari Stage** dengan cara menekan tombol **Delete**. *MovieClip* transisiMC tersebut tidak dibutuhkan di *Stage*, karena akan diletakkan ke layar menggunakan kode. Perhatikan teknik penambahan objek dengan **Linkage** pada bab sebelumnya. Setiap objek yang akan ditambahkan ke layar dengan menggunakan kode, wajib memiliki **Linkage**. Oleh karena itu *MovieClip* transisiMC membutuhkan **Linkage**.

Untuk menambahkan **Linkage** pada *MovieClip* transisiMC, **buka Library** dengan cara menekan **Ctrl+L**. Kemudian **klik kanan MovieClip transisiMC** dan pilih opsi **Properties**. Pada panel **Symbol Properties** tekan menu **Advanced** dan centang opsi **Export for Actionscript**, maka pada kolom *class* akan terisi dengan nama *class* (pada umumnya sama dengan nama *MovieClip* namun tanpa spasi). Nama *class* inilah yang disebut sebagai **Linkage**.



Gambar 164. menambahkan *Linkage*

4. Setelah *Linkage* ditambahkan, tahapan berikutnya adalah mengaktifkan efek transisi dengan menggunakan kode. **Klik *Frame 10 Layer kode***, buka panel **Action** dan tambahkan kode berikut :

```
mpi.setTransisi("mc", "transisiMC");
```

5. Jalankan aplikasi dengan menekan **Ctrl+Enter**, maka setiap kali perpindahan halaman akan menampilkan efek transisi tirai yang tertutup dan terbuka. Sistem kerja dari efek transisi *MovieClip* adalah transisi akan ditambahkan ke layar dan animasi akan berjalan. *MovieClip* transisiMC yang dibuat pada langkah 3 memiliki durasi 30 *Frame* (1 detik), dan tepat pada *Frame 15* tirai tertutup seutuhnya. Pada *Frame 15* tersebut (yaitu setengah dari durasi total animasi), *Frame* aktif akan dipindah menuju ke *Frame* berikutnya. Sehingga ketika tirai terbuka kembali *Frame* sudah berpindah, dan terjadilah efek pergantian halaman.

Dengan teknik yang sama, berbagai jenis transisi dapat dibuat dan ditambahkan sebagai efek transisi. Secara *default* pergantian halaman akan terjadi tepat setengah dari durasi *MovieClip* transisi, namun apabila titik pergantian halaman tidak berada di tengah, maka dapat ditambahkan parameter baru yaitu *Frame* tempat pergantian halaman terjadi dengan mengubah sedikit kode menjadi :

```
mpi.setTransisi("mc", "transisiMC", 25);  
//25 adalah keyframe tempat perpindahan halaman
```

Apabila proses penambahan *Linkage* tidak benar, atau *MovieClip* transisi belum ditambahkan *Linkage*, maka akan muncul pesan kesalahan pada panel **Compiler Error** sebagai berikut :

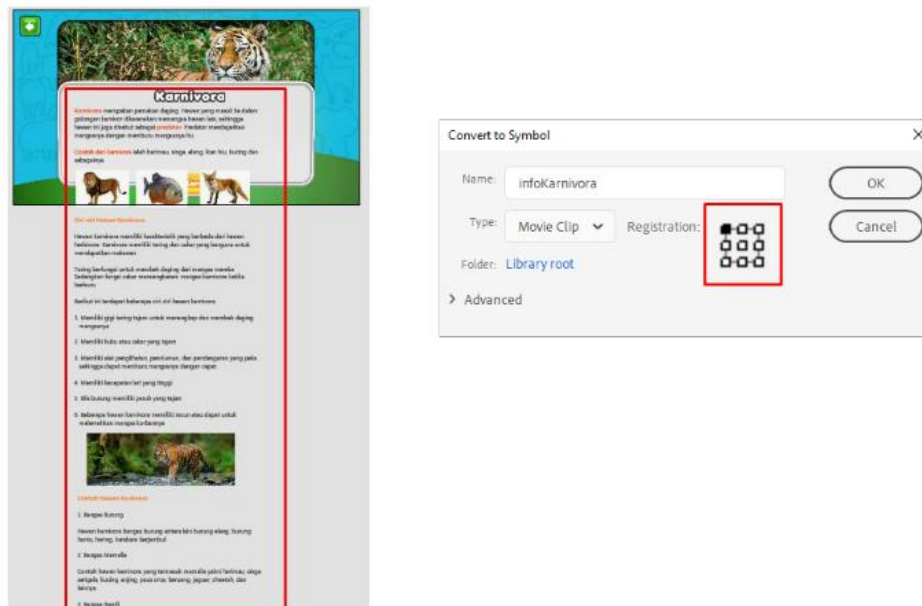
```
ReferenceError: Error #1065: Variable transisiMC is not defined.
at global/flash.utils::getDefinitionByName()
at MPI/transisi()
at MPI/doBtn()
```

Solusi atas permasalahan tersebut adalah dengan menambahkan *Linkage* pada *MovieClip* transisi dengan menjalankan langkah no 3 di atas.

6.6 Scroll konten

Pada penjelasan di bab sebelumnya, sebuah informasi dapat ditampilkan dalam format *scroll* ketika informasi yang ingin ditampilkan cukup banyak dan melebihi area layar yang tersedia. Pada langkah berikut akan ditambahkan konten *scroll* pada halaman “hewan karnivora” (*Frame 30*). Konten *Static Text* yang ada akan diganti dengan konten yang lebih detail, di dalamnya terdapat tulisan dan gambar. Perhatikan langkah-langkah berikut, untuk membuat konten *scroll* :

1. Buka kembali file (**tutorial 8 – Tebak Satwa**), dan file **MPI-component.fla**.
2. Klik **Frame 30 Layer konten** (halaman “hewan karnivora”). Ubah informasi yang sudah ada, tambahkan teks yang lebih banyak, atau gambar-gambar. Teks dapat ditulis secara terpisah, dan ditulis memanjang ke arah bawah. Tidak ada batasan ukuran untuk membuat informasi yang akan *discroll* (tidak masalah apabila konten keluar dari *Stage*).



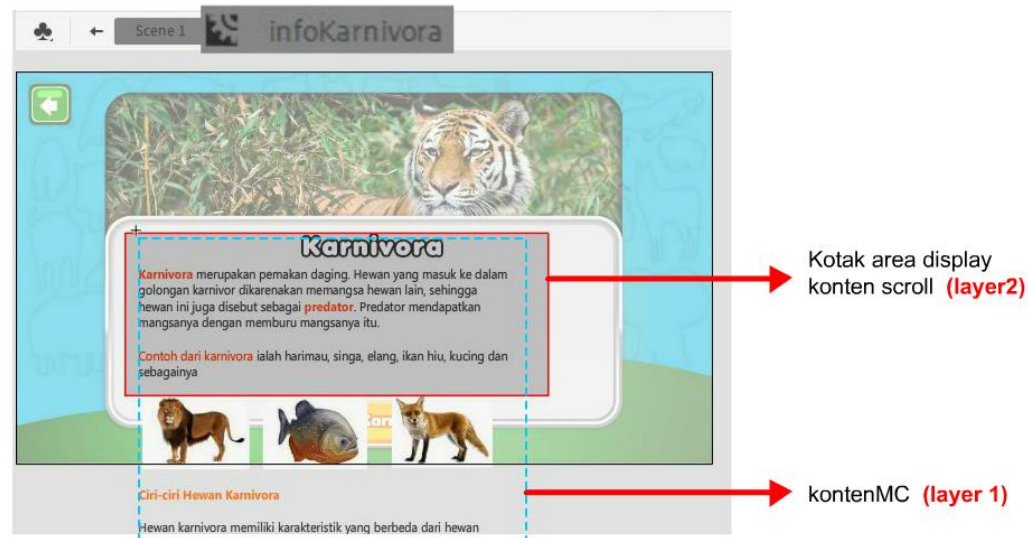
Gambar 165. membuat informasi untuk fitur *scroll*

Seleksi seluruh informasi yang akan dijadikan konten *scroll* (gambar dan teks), kemudian **convert menjadi MovieClip “informasi Karnivora”**. Sebelum menekan tombol OK pastikan **titik registrasi MovieClip** tersebut berada di **pojok kiri atas**.

Dobel klik *MovieClip* tersebut untuk mengeditnya. Di dalam mode edit, seleksi kembali seluruh teks dan gambar yang ada, kemudian **convert menjadi *MovieClip*** “**konten scroll karnivora**”. Pastikan pula **titik registrasi** berada di **pojok kiri atas**. Pada tahapan ini pastikan bahwa di dalam *MovieClip* “informasi karnivora” terdapat *MovieClip* “konten scroll karnivora” (*MovieClip* di dalam *MovieClip*).

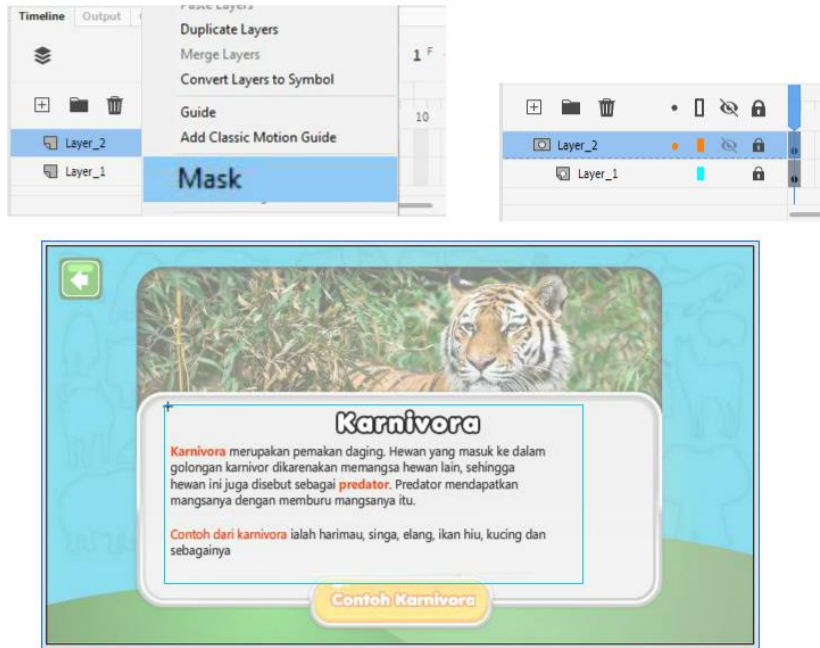
Klik *MovieClip* “konten scroll karnivora”, buka panel ***Properties*** dan tambahkan ***instance name*** “**kontenMC**”. *Instance name* untuk konten yang akan *discroll* wajib bernama “kontenMC” karena nama tersebut yang akan diakses oleh *MPI Component*, pemberian nama yang berbeda akan menyebabkan *error*.

Sebuah konten yang akan *discroll* pada umumnya memiliki ukuran yang cukup besar, sementara area yang akan ditampilkan relatif lebih kecil. Untuk mensiasati hal tersebut digunakan fitur ***masking***. Untuk melakukannya, **buatlah sebuah *Layer* baru (*Layer* 2)**, selanjutnya dengan menggunakan ***Rectangle Tools***, **buatlah kotak** seukuran area yang akan digunakan untuk *mendisplay* konten *scroll*.



Gambar 166. struktur *MovieClip* “informasi karnivora”

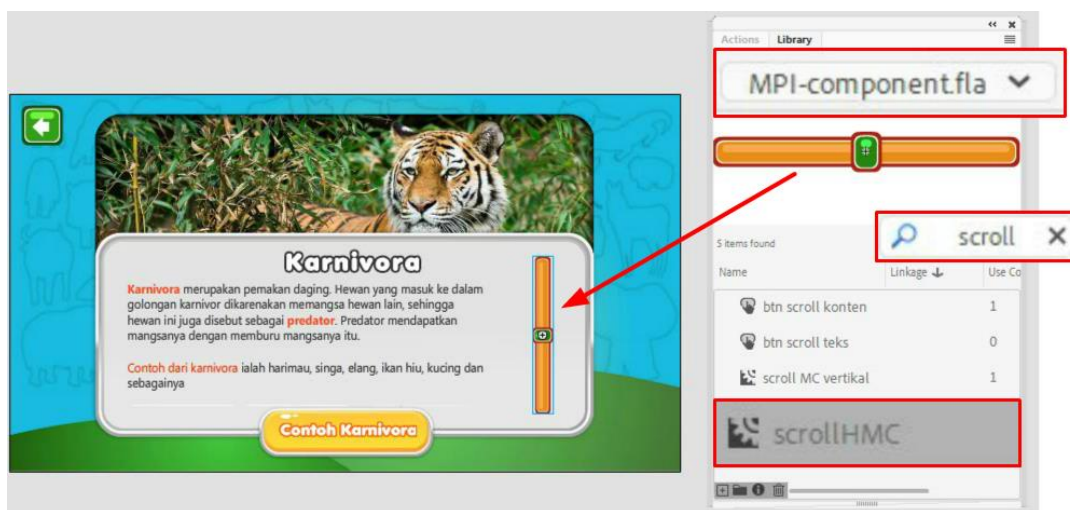
Untuk mengaktifkan fitur *masking*, **klik kanan *Layer* 2**, kemudian pilih opsi ***Mask***. Maka *Layer* 1 akan terdorong masuk, kontenMC akan terlihat sebatas area kotak yang telah dibuat di *Layer* 2, dan kotak di *Layer* 2 tidak akan terlihat. Konsep dasar fitur *masking* adalah, apapun yang ditutup oleh area *masking*, akan terlihat di layar.



Gambar 167. proses *masking*

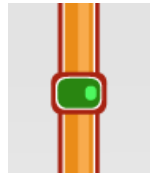
3. Keluar dari mode edit *symbol* dengan menekan **Ctrl+E**. Selain konten yang akan *discroll*, juga dibutuhkan elemen antarmuka untuk melakukan *scrolling* (disebut sebagai *scroller*). Dalam hal ini, *scroller* dapat diambil dari *Library file* MPI-component. Buka *Library file* MPI-Component, ketikkan "*scroll*" pada kolom *search*, dan akan didapatkan *MovieClip* "*scrollHMC*".

Drag *MovieClip* scrollHMC tersebut ke *Stage*, dan dengan menggunakan ***Transform Tool***, atur ulang posisinya (jadikan tegak).



Gambar 168. menambahkan *scroller*

Untuk memutar *MovieClip* scrollHMC menjadi vertikal perlu diperhatikan arah putaran. *MovieClip* harus diputar searah jarum jam agar *scroller* bekerja dengan baik (tidak terbalik). Untuk memudahkan, perhatikan cahaya atau highlight pada tombol hijau. Pastikan warna hijau muda berada di sebelah kanan.



Gambar 169. posisi *MovieClip* scrollHMC saat vertikal

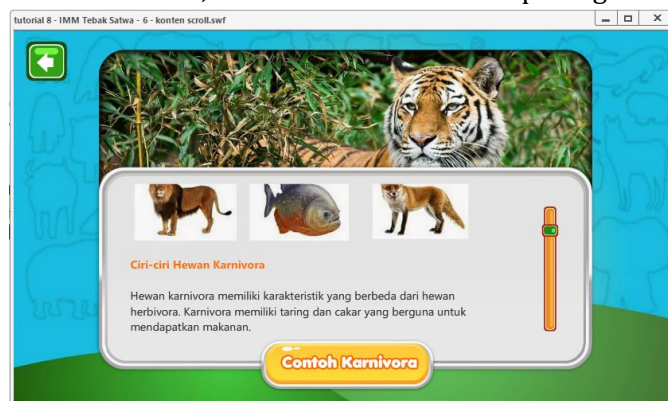
Apabila bentuk atau warna dari *MovieClip* scrollHMC tersebut kurang sesuai, maka dapat didobel klik dan diedit sesuai kebutuhan. Dalam mengedit elemen yang berasal dari *Library* MPI-Component, seluruh elemen visual dapat diedit sesuai kebutuhan, akan tetapi struktur *Layer* dan penamaan *instance name* di dalam *MovieClip* tidak perlu diubah.

4. Tahapan berikutnya adalah menambahkan ***instance name***. Klik ***MovieClip* info karnivora**, kemudian **tambahkan *instance name* "infoKarnivora"**. Klik ***MovieClip* scrollHMC** kemudian **tambahkan *instance name* "scrollMC"**.
5. Untuk mengaktifkan fitur *scroll* diperlukan penambahan kode. Klik **Frame 30 Layer kode**, buka panel **Action** dan tambahkan kode berikut :

```
mpi.scroll("scrollMC", 0, scrollKarnivora);

function scrollKarnivora() {
    mpi.scrollMC("infoKarnivora", "y", scrollMC.val);
}
```

6. Jalankan aplikasi dengan menekan **Ctrl+Enter**. Klik tombol Jenis hewan > Karnivora, apabila tidak ada kesalahan, maka fitur *scroll* akan dapat digunakan.



Gambar 170. hasil penambahan fitur *scroll*

6.6.1 Penjelasan Kode

```
mpi.scroll("scrollMC", 0, scrollKarnivora);
```

Kode `mpi.scroll` digunakan untuk mengaktifasi *MovieClip scroller*. Parameter yang dimasukkan adalah *instance name MovieClip scroller*, dalam hal ini adalah "scrollMC". Angka 0 merupakan nilai awal dari dimulainya proses *scrolling*, dalam hal ini adalah dimulai dari atas. Parameter `scrollKarnivora` merupakan fungsi yang akan dijalankan ketika proses *scrolling* dilakukan, sehingga kapanpun *MovieClip scroller* digeser, maka fungsi akan dijalankan. Nilai pergeseran dari proses *scrolling* adalah rentang 1-100 dan dapat diambil dengan kode `scrollMC.val`.

```
function scrollKarnivora(){  
    mpi.scrollMC("infoKarnivora", "y", scrollMC.val);  
}
```

Fungsi `scrollKarnivora` merupakan perintah yang akan dijalankan akibat aktivasi kode `mpi.scroll`. Di dalamnya terdapat sebuah kode yaitu `mpi.scrollMC`. Kode ini akan menggeser *MovieClip* kontenMC yang ada di dalam *MovieClip* "infoKarnivora". Pergeseran akan dilakukan terhadap sumbu "y", dan pergeseran yang dilakukan sebesar nilai `scrollMC` dalam format prosentase (1-100).

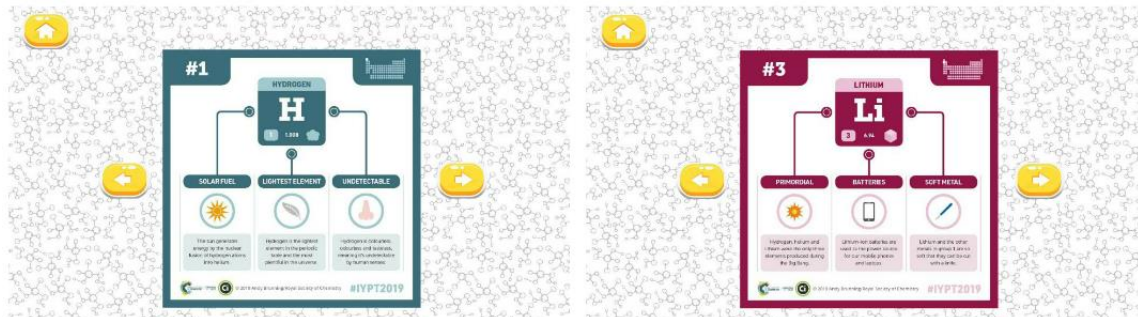
6.7 Galeri

Salah satu fitur untuk menampilkan informasi sejenis dalam jumlah yang banyak adalah dengan menggunakan fitur galeri. Sebagai contoh, untuk menampilkan informasi tentang contoh hewan karnivora, maka akan ada beberapa data sejenis masing-masing hewan karnivora seperti harimau, singa, ikan hiu, burung elang dan sebagainya. Data sejenis tersebut dapat ditampilkan dalam format galeri yang dapat dengan mudah digeser oleh pengguna. Sebagai contoh, multimedia interaktif "I am mammals" karya Catherine Firly (2022) menampilkan contoh hewan-hewan mamalia dalam format galeri yang dapat digeser ke kanan dan ke kiri menggunakan tombol.



Gambar 171. Contoh fitur galeri dalam multimedia interaktif (Firly, 2022)

Contoh lain adalah multimedia interaktif “Belajar kimia”, yang menampilkan unsur-unsur kimia secara berurutan dalam bentuk infografis. Pengguna dapat menggeser unsur-unsur kimia yang ada dan ditampilkan dengan tata letak yang seragam.



Gambar 172. fitur galeri pada multimedia interaktif “Belajar kimia” (Wibawanto, 2022)

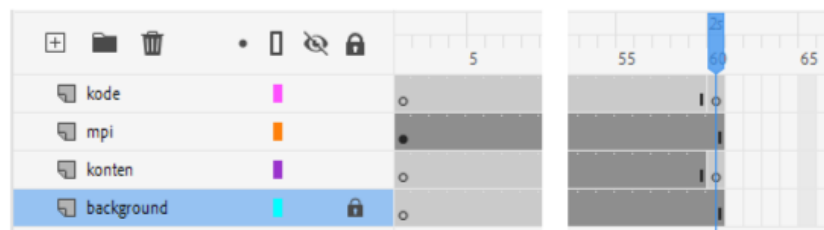
Sebelum menambahkan fitur galeri ke dalam proyek multimedia interaktif “Tebak Satwa”, terdapat persiapan yang harus dilakukan, yaitu membuat konten foto binatang karnivora dengan nama *file* yang dibuat berurutan. Maksud dari penamaan *file* yang berurutan adalah untuk mempermudah proses impor.



Gambar 173. *file* gambar hewan karnivora

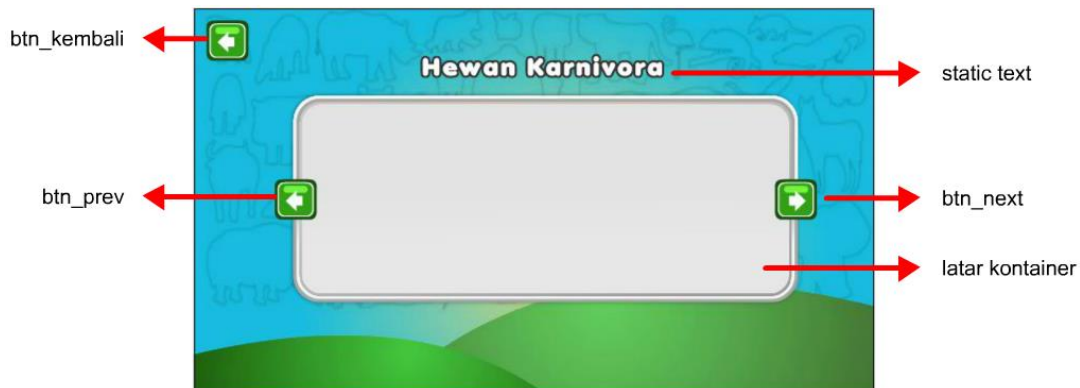
Fitur galeri untuk menampilkan contoh hewan karnivora akan diletakkan pada *Frame* 60. Perhatikan langkah-langkah berikut untuk menambahkan fitur galeri :

1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**).
2. Untuk menambahkan halaman baru, dapat ditambahkan *Frame* dan *Blank Keyframe* pada *Frame* 60. **Tambahkan Frame** pada **Frame 60 Layer background** dan **Frame 60 Layer mpi** (penambahan *Frame*, karena tidak membutuhkan perubahan konten). Selanjutnya **tambahkan Blank Keyframe** pada **Frame 60 Layer konten** dan **Frame 60 Layer kode** (penambahan *Blank Keyframe*, untuk menyediakan *Frame* kosong untuk konten baru).



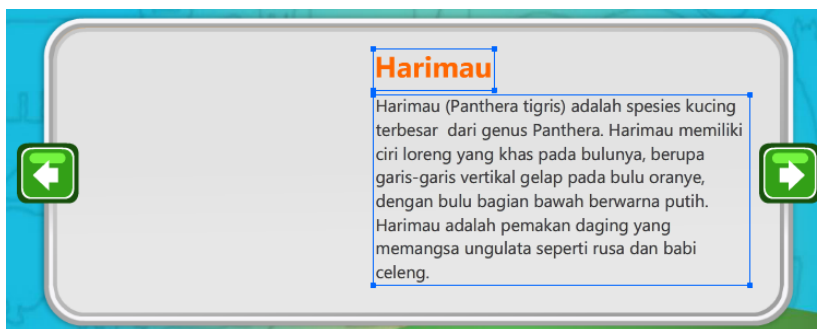
Gambar 174. struktur *Timeline* pada *Frame* 60

3. **Klik *Frame 60 Layer konten*** selanjutnya buatlah sebuah konten untuk halaman contoh hewan karnivora. Pada halaman ini tambahkan beberapa tombol yang sudah ditambahkan ***instance name*** serta sebuah latar kontainer sebagai tempat meletakkan informasi berupa gambar dan teks.



Gambar 175. konten pada halaman “contoh hewan karnivora”

4. Untuk membuat konten galeri yang dapat digeser, buatlah ***Static Text*** yang berisi informasi tentang contoh hewan karnivora yang sesuai dengan gambar pertama, dalam contoh ini adalah harimau. *Static Text* ditambahkan pada sebelah kanan, karena pada bagian kiri nantinya akan ditambahkan foto binatang. Selain itu hal yang perlu diperhatikan adalah konsistensi penggunaan huruf. Janganlah menggunakan huruf yang beraneka jenis bentuk, warna dan ukurannya. Tentukan di awal huruf yang akan digunakan untuk judul, sub judul dan huruf untuk paragraf/kalimat.

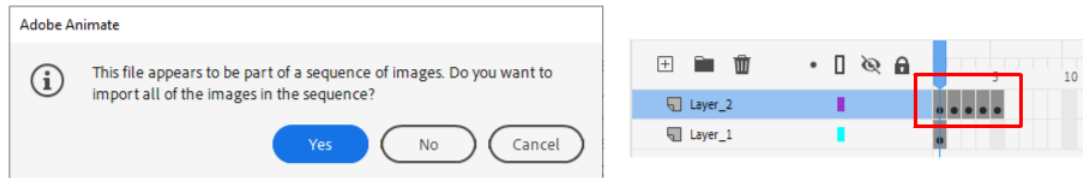


Gambar 176. penambahan informasi

Seleksi *Static Text* tersebut, kemudian ***convert*** menjadi ***MovieClip*** “contohKarnivoraMC”. **Dobel klik *MovieClip*** tersebut untuk mengeditnya. Pada tahapan berikutnya akan ditambahkan gambar.

Pada mode edit *symbol*, **buatlah sebuah *Layer* baru (*Layer 2*)**. **Klik *Frame 1 Layer 2*** kemudian ***import file*** “karnivora_1” ke ***Stage***. Apabila *file* yang diimport memiliki nama yang berurutan (lihat gambar di atas), maka akan muncul sebuah panel yang

menanyakan “apakah *file* akan diimport semua dalam format sequence. Pilih opsi **Yes**, maka akan terbentuk struktur *Timeline* secara **Frame by Frame** yang masing-masing *Frame* diisi oleh sebuah gambar sesuai dengan nomor urutnya. Metode ini akan menghemat waktu, karena struktur *Timeline* sudah dibentuk secara otomatis oleh sistem.



Gambar 177. mengimpor *file* dengan nama berurutan

Klik **Frame 1 Layer 1**, saat ini dimungkinkan posisi gambar tidak sesuai dengan posisi *Static Text*. Oleh karena itu, **sesuaikan** terlebih dahulu **posisi teks terhadap gambar**. Penggeseran yang lebih efektif adalah menggeser *Static Text*, hal ini dikarenakan gambar sudah tertata secara *Frame by Frame*.



Gambar 178. menyesuaikan ulang posisi teks terhadap gambar

Pada saat ini posisi *MovieClip* contohKarnivoraMC tidak sesuai dengan kontainer yang berada di *Frame 60 Stage* utama. Untuk itu keluar dari mode edit dengan menekan tombol **Ctrl+E** lalu **sesuaikan posisi MovieClip** terhadap kontainer.



Gambar 179. menyesuaikan posisi *MovieClip* terhadap kontainer

Selanjutnya **dobel klik** *MovieClip* tersebut untuk mengedit kembali konten *Frame* berikutnya. **Klik Frame 2 Layer 1** tambahkan **Keyframe**, selanjutnya **edit** informasi yang ada, sesuaikan dengan gambar di *Frame 2*.



Gambar 180. menambahkan informasi pada *Frame 2 MovieClip* contohKarnivoraMC

Klik Frame 3 Layer 1 tambahkan **Keyframe**, selanjutnya **edit** informasi yang ada, sesuaikan dengan gambar di *Frame 3*. Lakukan hal yang sama sampai seluruh informasi yang ada selesai dibuat dalam format *Frame by Frame*. Dalam contoh ini terdapat 5 buah informasi.



Gambar 181. penambahan informasi secara *Frame by Frame*

5. Keluar dari mode edit dengan menekan tombol **Ctrl+E**. Selanjutnya **tambahkan instance name** pada *MovieClip* "contohKarnivoraMC". Pastikan juga *instance name* pada tombol-tombol yang ada sudah ditambahkan.
6. Untuk penambahan kode, terlebih dahulu tombol **btn_contohkarnivora** pada halaman hewan karnivora (*Frame 30*) diaktifkan. **Klik Frame 30 Layer kode** kemudian tambahkan kode berikut :

```
mpi.click("btn_contohKarnivora", 60);
```

Selanjutnya untuk menambahkan fitur galeri, klik **Frame 60 Layer kode** dan ketikkan kode berikut :

```
mpi.click("btn_kembali", 30);
mpi.click("btn_next", geserGaleri, 1);
mpi.click("btn_prev", geserGaleri, -1);
contohKarnivoraMC.stop();

function geserGaleri(num):void{
    mpi.geser("contohKarnivoraMC", num);
}
```

7. Simpan dan jalankan aplikasi. Ujicoba sistem galeri dengan menekan tombol ke kanan dan ke kiri, apabila tidak ada kesalahan maka fitur galeri akan bekerja dengan memindah halaman di dalam *MovieClip* contohKarnivoraMC.

6.7.1 Penjelasan Program

Pada penambahan fitur galeri di atas, terdapat beberapa kode baru :

```
mpi.click("btn_next", geserGaleri, 1);
```

Fungsi `click` tidak hanya digunakan untuk berpindah ke *Frame* lain, namun juga dapat digunakan untuk menjalankan fungsi tertentu saat sebuah tombol diklik. Pada contoh tersebut ketika `btn_next` diklik maka fungsi `geserGaleri` akan dijalankan dengan mengirimkan angka 1 ke fungsi tersebut. Angka ini akan digunakan untuk menggeser *Frame* sebanyak 1 satuan. Perhatikan pada kode `click` berikutnya, angka yang dikirim adalah -1 yang berarti memundurkan 1 *Frame*.

```
mpi.geser("contohKarnivoraMC", num);
```

Kode `geser` digunakan untuk memindah *Frame* sebuah *MovieClip* sebanyak parameter `num`. Dalam contoh ini ketika tombol `btn_next` ditekan, maka angka 1 akan dikirim sehingga *MovieClip* contohKarnivoraMC akan berpindah maju satu *Frame*.

6.7.2 Menambahkan Efek Pergeseran pada Galeri

Untuk menghasilkan tampilan yang dinamis, dapat ditambahkan efek pergeseran pada fitur galeri. Pada saat buku ini ditulis, terdapat 2 efek pergeseran yaitu "fade" dan "slide". Untuk menambahkan efek pergeseran pada galeri, perhatikan langkah berikut :

1. Buka panel **Library (Ctrl+L)**, kemudian tambahkan **Linkage** pada *MovieClip* contohKarnivoraMC, bedakan nama *Linkage* dengan *instance name*. Sebagai contoh

nama *Linkage* dari *MovieClip* contohKarnivoraMC adalah **karnivoraMC** sedangkan *instance namenya* sama dengan nama *MovieClip*.

Apabila digunakan nama *Linkage* dan *instance name* yang sama, maka akan terjadi kesalahan karena terdapat dua identitas yang sama dalam sebuah aplikasi.

2. **Klik Frame 60 Layer kode**, kemudian ubah sedikit kode geser menjadi sebagai berikut:

```
mpi.geser("contohKarnivoraMC", num, "fade");
```

3. Jalankan kembali dengan menekan **Ctrl+Enter**.



Gambar 182. penambahan *Linkage* dan hasil penambahan efek “fade”

Untuk penambahan efek “slide” diperlukan latar yang *solid*, misalnya pada bagian latar belakang ditambahkan kotak. Hal ini dikarenakan efek *slide* menggunakan teknik *masking*, sehingga membutuhkan area *masking* yang tertutup oleh objek *solid*.

Sampai dengan tahapan ini, multimedia interaktif sudah dapat dijalankan dengan sistem navigasi dasar. Dengan teknik yang sama, halaman-halaman selanjutnya dapat ditambahkan. Konten halaman dapat ditampilkan secara statis, menggunakan teknik *scroll* atau menggunakan sistem galeri. Penambahan-penambahan animasi dalam format *MovieClip* juga dapat ditambahkan untuk meningkatkan daya tarik aplikasi.

Pada tahapan selanjutnya akan dijelaskan teknik penambahan suara dan video untuk melengkapi fitur-fitur yang ada dalam aplikasi multimedia interaktif yang dibuat.

BAB 7

Programming Phase

Suara dan Video

7.1 Penambahan Suara

Suara adalah sebuah syarat wajib bagi sebuah aplikasi multimedia interaktif. Dalam aplikasi Adobe Animate, suara dapat ditambahkan dengan 2 cara, yaitu :

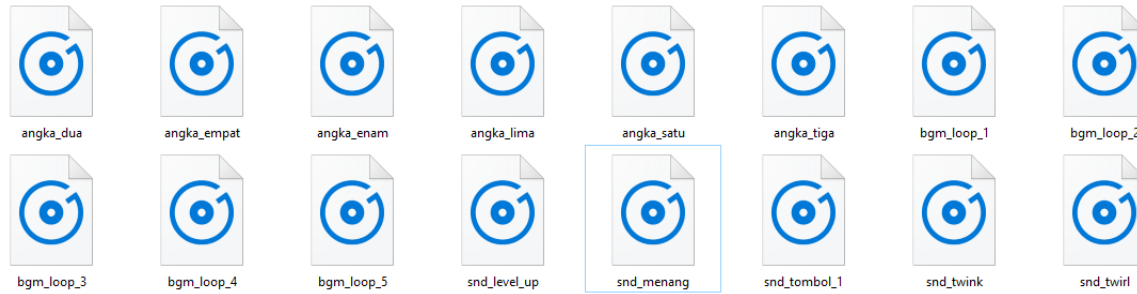
1. impor suara ke *Stage* dan menempatkannya pada sebuah *Keyframe*. Teknik ini sangat mudah dilakukan, namun memiliki kelemahan yaitu suara tidak dapat diatur secara dinamis. Sebagai contoh apabila pengguna ingin mematikan suara atau mengatur volume suara, maka pada teknik “manual” seperti ini tidak dapat dilakukan.
2. Impor suara ke *Library* dan menambahkannya aplikasi menggunakan kode. Teknik ini melibatkan pemrograman, dan dengan penggunaan *MPI Component* kode yang digunakan relatif sedikit. Teknik ini memiliki keuntungan yaitu suara dapat diatur volume, menyalakan dan mematikan dengan dinamis.

Adobe Animate mendukung beberapa format *file* suara yang dapat diimport ke dalam aplikasi, di antaranya yang paling umum digunakan adalah *file* bertipe WAV dan MP3. Dalam pengembangan multimedia interaktif, penggunaan suara perlu dirancang sejak awal untuk mengetahui kebutuhan suara, sebagai contoh dalam multimedia interaktif “Tebak Satwa” yang sedang dikembangkan diperlukan beberapa suara sebagai berikut :

Tabel 7.1 List suara yang dibutuhkan

No	Halaman	Elemen Audio	File Audio
1	Semua halaman utama	Suara Latar	bgm_loop_4.wav
		Suara tombol	snd_tombol_1.mp3
2	Jenis Hewan	Suara hewan	snd_harimau, snd_Ayam, snd_kucing dst.
3	Halaman Permainan Kuis	Suara soal muncul	snd_level_up.mp3
		Suara jawaban benar	snd_twink.mp3
		Suara jawaban salah	snd_twirl.mp3
		Suara kuis selesai	snd_menang.mp3

Dari tabel list suara di atas, didapatkan gambaran beberapa *file* suara yang dibutuhkan untuk aplikasi. Selanjutnya *file* suara dapat dibuat sendiri atau mengunduh *file* suara yang tersedia secara legal di beberapa situs penyedia suara. Kumpulkan *file-file* suara tersebut ke dalam sebuah *folder* untuk mempermudah proses import ke dalam aplikasi.

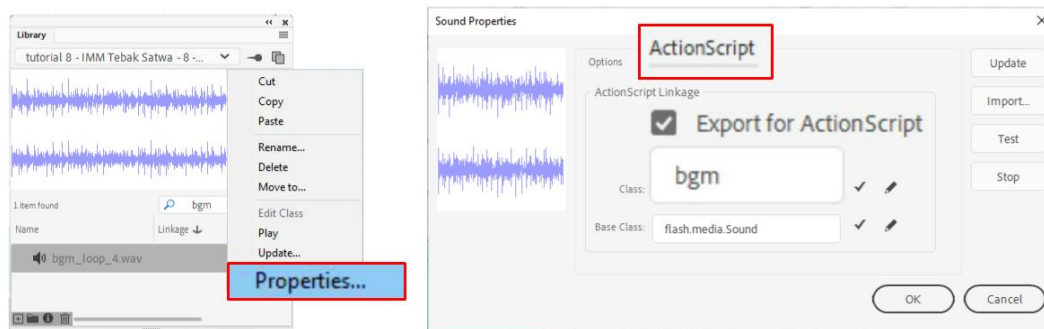


Gambar 183. persiapan *file* suara

Perhatikan tahapan berikut untuk menambahkan suara ke dalam aplikasi:

1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**).
2. Suara yang akan ditambahkan terlebih dahulu adalah suara latar. **Impor *file* suara** ke **Library** (Import > Import to Library), dalam contoh ini diimport *file* **bgm_loop_4.wav**.

Berbeda dengan objek lain yang memiliki fisik (dapat diklik, digeser dan sebagainya), suara tidak memiliki bentuk fisik, sehingga tidak dapat diberikan *instance name*. Untuk mengakses suara dari *Library* dibutuhkan **Linkage**. Untuk menambahkan *Linkage* ke objek suara, **buka panel Library**, pilih opsi **Properties**. Berbeda dengan *Properties symbol MovieClip*, pada panel *Sound Properties* tidak ada tombol *advanced*. **Klik tab *Actionscript*** kemudian centang opsi **Export for Actionscript** dan ubah nama *class* menjadi **"bgm"**.



Gambar 184. menambahkan *Linkage* pada suara

3. Untuk mengaktifkan suara perlu dilakukan penambahan kode, dan penambahan kode dapat diletakan pada *Frame 10* (halaman judul) , sehingga suara akan muncul pada saat pengguna memasuki halaman judul. **Klik *Frame 10 Layer* kode**, kemudian tambahkan kode berikut :

```
mpi.suara("bgm", "loop");
```

4. Jalankan aplikasi, maka akan didapatkan suara ketika masuk ke halaman judul.

Pada kode `mpi.suara` terdapat beberapa parameter yaitu “bgm” yang merupakan *Linkage* dari suara, serta “loop” untuk menjalankan suara secara berulang. Parameter ini dapat diisi dengan “play” untuk menjalankan suara satu kali saja, dan “stop” untuk menghentikan suara. Untuk lebih detailnya berikut variasi kode `mpi.suara` :

```
mpi.suara("bgm"); //menjalankan suara bgm satu kali
mpi.suara("bgm", "play"); //menjalankan suara bgm satu kali
mpi.suara("bgm", "loop"); //menjalankan suara bgm secara berulang
mpi.suara("bgm", "loop", 40); //suara bgm berulang dengan volume 40%
mpi.suara("bgm", "stop"); //menghentikan suara bgm
mpi.suaraAktif = false; //status suara, true = aktif, false = mati
```

7.2 Menambahkan Suara Tombol

Salah satu efek suara yang ada dalam aplikasi adalah suara tombol ketika ditekan. Cara paling mudah untuk melakukannya adalah dengan mengimpor suara dan meletakkannya pada *Frame down* sebuah tombol, namun suara tidak dapat dimatikan. Cara yang lebih efektif adalah dengan menggunakan kode `mpi.suaraTombol`. Perhatikan langkah berikut :

1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**).
2. **Impor** *file* suara tombol ke **Library**. Kemudian **tambahkan Linkage**, sebagai contoh digunakan *file* “snd_tombol_1.wav” dan ditambahkan *Linkage* “snd_tombol”.
3. **Klik Frame 10 Layer kode**, kemudian tambahkan kode berikut :

```
mpi.suaraTombol("semua", "snd_tombol");
```

4. **Jalankan** kembali aplikasi dan ujicoba dengan menekan salah satu tombol aktif.

Kode `mpi.suaraTombol` akan mengaktifkan suara saat tombol ditekan. Pada fungsi tersebut terdapat parameter “semua” yang berarti semua tombol yang ada akan memiliki efek suara ditekan yang sama yaitu “snd_tombol”. Untuk mengaktifkan efek tombol tertentu saja, maka parameter “semua” dapat diganti dengan *instance name* tombol yang dimaksud, misal :

```
//suara khusus untuk tombol btn_harimau
mpi.suaraTombol("btn_harimau", "snd_harimau");
```

Dengan penambahan kode tersebut, ketika suara dinonaktifkan, maka suara tombol juga akan non aktif.

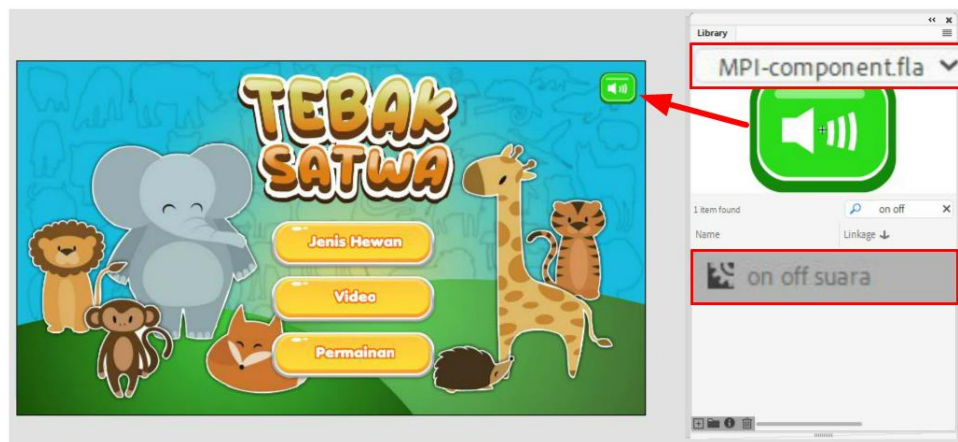
7.3 Menambahkan Fitur *On-Off* Suara

Pada setiap aplikasi multimedia interaktif diperlukan sebuah opsi untuk mengaktifkan dan menonaktifkan suara. Hal ini ditujukan untuk memberikan pengalaman pengguna (*User Experience*) yang baik. Beberapa pengguna menginginkan adanya suara latar seperti musik, namun beberapa pengguna lainnya merasa terganggu dengan keberadaan musik. Oleh karena itu diperlukan sebuah fitur yang membebaskan pengguna dalam menentukan pilihannya terkait aktif tidaknya musik dalam aplikasi.

Menambahkan fitur *onOff* suara dapat ditampilkan dalam sebuah *Toggle Button* yang dimunculkan secara langsung pada halaman utama, atau dimasukkan secara khusus ke dalam halaman pengaturan aplikasi. Pada contoh ini akan dilakukan penambahan *toggle button* untuk *onOff* suara pada halaman judul. Perhatikan langkah-langkah berikut :

1. Buka kembali file (**tutorial 8 – Tebak Satwa**) dan file *MPI-component fla*.
2. **Klik Frame 10 Layer konten** pada halaman judul ini akan ditambahkan sebuah tombol *onOff* suara. Untuk membuat sebuah tombol *onOff* diperlukan sebuah *MovieClip* yang memiliki 2 *Keyframe*. *Keyframe* pertama berisi tombol dalam kondisi *on*, dan *Keyframe* kedua berisi gambar dalam kondisi *off*.

Pada contoh ini digunakan *MovieClip* yang berasal dari *Library MPI-component*. **Buka Library MPI-component**, kemudian *drag MovieClip on off suara* ke *Stage*. Tempatkan di pojok kanan atas.



Gambar 185. menambahkan *onOff* suara

Apabila tampilan *MovieClip onOff* suara tersebut kurang sesuai dapat dilakukan perubahan dengan cara **dobel klik** *MovieClip*. Pada dasarnya di dalam *MovieClip* tersebut terdapat grafis kondisi *on* pada *Frame 1* dan grafis posisi *off* pada *Frame 2*. Lakukan pengeditan sesuai kebutuhan.



Gambar 186. struktur *MovieClip onOff* suara

3. **Tambahkan *instance name* “btn_onOff”** pada *MovieClip* tersebut. Pemberian nama instansi pada dasarnya fleksibel, meskipun simbol *on off* suara adalah *MovieClip*, dapat diberikan *instance name* berawalan “btn”. Hal ini tidak jadi masalah, karena kode mengenali sebuah nama berdasarkan satu kesatuan kata.
4. Untuk mengaktifasi fitur *onOff* suara tersebut **klik *Frame 10 Layer kode***, kemudian tambahkan kode berikut :

```
mpi.atursuara("btn_onOff");
```

5. Jalankan aplikasi dan ujicoba tombol untuk mengatur kondisi suara aplikasi.

Kode `mpi.atursuara` secara otomatis akan memposisikan *Frame MovieClip* “btn_onOff” pada kondisi `mpi.suaraAktif`. Ketika suara dimatikan, dan pengguna berpindah ke *Frame* lain, lalu kembali ke halaman judul suara tetap akan dalam kondisi *off*. Demikian pula sebaliknya.

7.4 Menambahkan Fitur Pengaturan Volume Suara

Fitur audio yang acapkali ditambahkan dalam suatu aplikasi selain *onOff* adalah fitur pengaturan volume suara. Pada beberapa aplikasi pengaturan volume suara ditampilkan dalam format *popup* atau ditampilkan dalam halaman terpisah (halaman *setting*). Pada contoh berikut, akan dilakukan penambahan fitur pengaturan volume suara pada halaman terpisah :

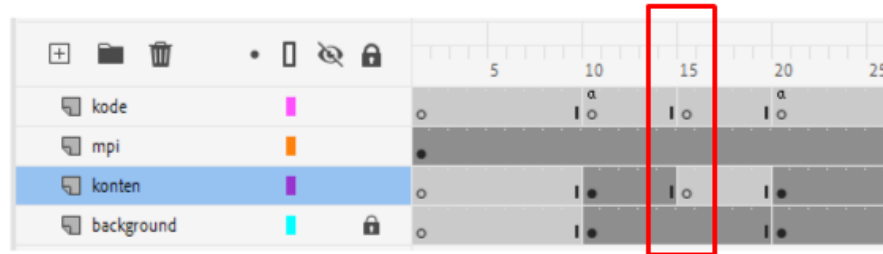
1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**) dan *file* MPI-component fla.
2. **Klik *Frame 10 Layer konten*** pada halaman judul tersebut tambahkan sebuah **tombol “btn_setting”** pada pojok kiri atas. Tombol dapat dibuat sendiri atau mengambil dari *Library file* MPI-component. **Tambahkan *instance name* “btn_setting”** pada tombol tersebut.



Gambar 187. menambahkan tombol “btn_setting”

- Halaman pengaturan (*setting*) akan disisipkan pada *Frame 15*. Salah satu keuntungan penggunaan kelipatan 10 antar halaman multimedia interaktif adalah dapat menyisipkan halaman-halaman baru di antara halaman utama. Pada contoh ini, halaman *setting* akan ditambahkan di *Frame 15*.

Seperti halnya penambahan halaman pada bab sebelumnya, penambahan halaman pada *Frame 15* membutuhkan **Blank Keyframe** pada **Frame 15 Layer kode** dan **Layer konten**.



Gambar 188. struktur *Timeline* pada *Frame 15*

- Untuk menambahkan konten halaman *setting*, **klik *Frame 15 Layer konten***, kemudian tambahkan aset visual di antaranya : tombol *home*, latar belakang untuk kontainer *setting*, *Static Text*, dan *scrollHMC* (dari *Library MPI-Component*). **Tambahkan *instance name*** pada "*scrollMC*" dan "*btn_home*".



Gambar 189. aset visual pada halaman *setting*

- Klik *Frame 10 Layer kode*** kemudian tambahkan kode berikut :

```
mpi.click("btn_setting", 15);
```

Selanjutnya **klik *Frame 15 Layer kode*** dan ketikkan kode berikut :

```
mpi.click("btn_home", 10);
mpi.suara("bgm", "loop");
```

```

mpi.scroll("scrollMC", mpi.volumeSuara, aturVolume);

function aturVolume() {
    mpi.suara("bgm", "loop", scrollMC.val);
}

```

6. **Jalankan** aplikasi dan ujicoba halaman *setting* suara, pastikan pengaturan volume suara menggunakan scrollerMC dapat dilakukan.

Penggunaan kode untuk mengatur volume suara di atas, pada dasarnya merupakan perpaduan antara kode `scroll` dan kode `suara.mpi.scroll` akan menjalankan fungsi `aturVolume` yang di dalamnya mengirimkan nilai *scrolling* (`scrollMC.val`) sebagai volume suara.

7.5 Penambahan Video

Elemen multimedia interaktif berikutnya yang secara umum ditambahkan ke dalam aplikasi adalah video. Video merupakan elemen multimedia yang memberikan informasi dalam format audio visual, sehingga mempermudah pengguna dalam memahami suatu informasi tertentu. Video dalam multimedia interaktif perlu dibuat terlebih dahulu secara terpisah dalam format MP4 atau FLV. Pengembangan video dapat dilakukan dengan menggunakan kamera (teknik *live shot*) maupun dengan menggunakan teknik animasi, baik itu animasi 2D maupun animasi 3D.

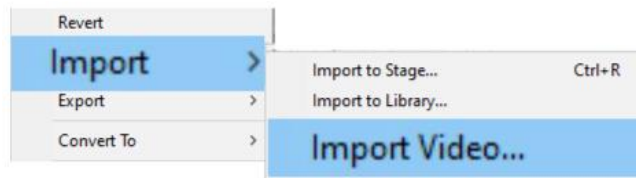
Aplikasi Adobe Animate mendukung penambahan video ke dalam sebuah proyek. Dalam proses penambahan video, terdapat 3 teknik yang secara umum dapat dilakukan, yaitu :

1. Membuka video secara eksternal. Pada teknik ini video diletakkan di luar aplikasi. Keuntungan dari teknik ini adalah ukuran *file* proyek tidak akan bertambah karena video berada di luar. Perlu dipahami bahwa *file* video pada umumnya memiliki ukuran yang besar, penambahan video ke dalam aplikasi secara langsung akan meningkatkan ukuran *file* akhir secara signifikan. Kelemahan dari metode ini, apabila pengaturan *folder* sumber *file* video tidak baik, maka video tidak dapat ditampilkan ketika aplikasi dijalankan.
2. Menambahkan *file* video ke dalam proyek dengan opsi *Embed*. Keuntungan dari teknik ini adalah *file* video telah berada di dalam proyek sehingga tidak perlu khawatir video gagal dibuka (sebagaimana sering terjadi pada teknik pertama). Selain itu, video dapat dimasukkan ke dalam *MovieClip*, sehingga pengaturannya mirip dengan mengatur *MovieClip*. Kelemahan dari metode ini adalah membengkaknya ukuran *file* proyek dan ukuran final *file* aplikasi karena terjadi penambahan sebesar ukuran video.
3. *Streaming* video dari sumber online. Cara ini berarti aplikasi akan membuka video yang telah diunggah ke suatu *server*. Teknik ini memiliki keuntungan ukuran *file* yang tidak bertambah, serta kesalahan *folder* tidak akan terjadi apabila memasukkan *link* video dengan benar. Namun kelemahan teknik ini adalah membutuhkan perijinan *server*

secara khusus serta membutuhkan koneksi internet setiap kali mengakses halaman yang memiliki video.

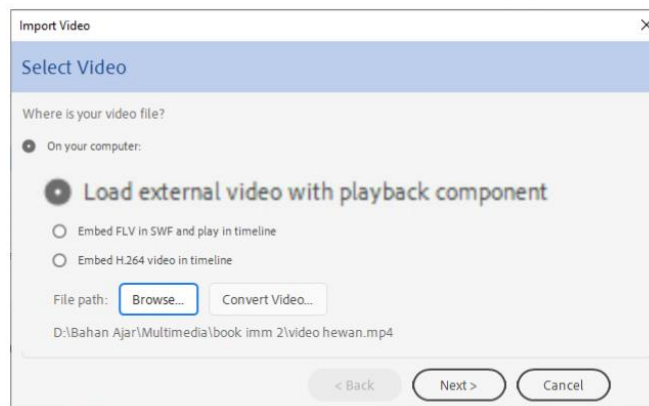
Pada contoh di buku ini digunakan teknik pertama, teknik tersebut lebih mudah untuk difahami bagi pengembang pemula, serta tidak membutuhkan pengkodean yang lebih rumit jika dibandingkan dengan kedua teknik lainnya. Perhatikan langkah-langkah berikut untuk menambahkan video ke dalam proyek multimedia interaktif “Tebak Satwa”.

1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**) dan *file* MPI-component fla.
2. Video akan ditambahkan pada halaman video , yaitu ketika pengguna menekan tombol video. Halaman ini akan dibuat pada *Frame* 70. Oleh karena itu **klik Frame 70 Layer kode** dan **Layer konten** , tambahkan **Blank Keyframe**. **Klik Frame 70 Layer mpi** dan **Layer background**, kemudian tambahkan **Frame**.
3. **Klik Frame 70 Layer konten**, kemudian tambahkan sebuah **tombol** dengan *instance name* “**btn_home**”.
4. Untuk mengimport video, **klik menu Import > Import Video**. Pastikan *file* video yang akan diimport berada dalam satu *folder* dengan *file* proyek yang sedang dibuat.



Gambar 190. menu import video

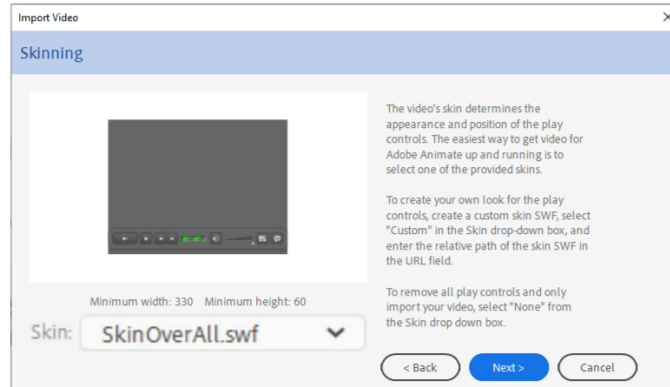
Pada panel import video, pilih opsi **Load external....** Kemudian tekan tombol **Browse** untuk mencari *file* video yang akan ditambahkan ke dalam aplikasi. Apabila tidak ada masalah dengan video yang akan diimport maka tahapan dapat dilanjutkan dengan menekan tombol **next**.



Gambar 191. memilih *file* video

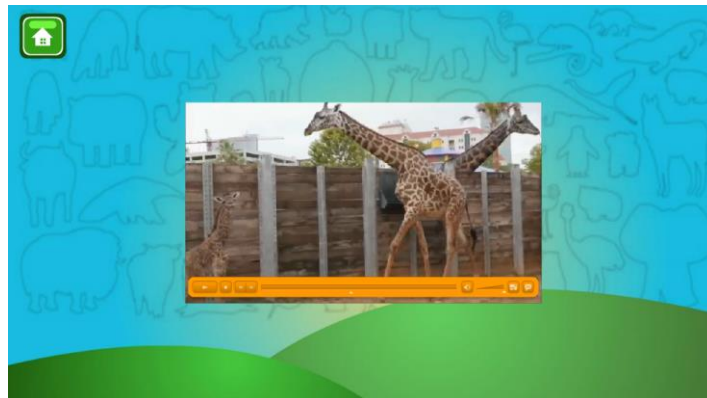
Tahapan berikutnya adalah pemilihan **skin**. *Skin* adalah sebuah antarmuka yang digunakan sebagai *video player*, di dalamnya terdapat beberapa fitur seperti tombol

play, stop, pause, seek dan sebagainya. Pemilihan *skin* ditujukan untuk mempermudah penggunaan video, namun apabila ingin mengembangkan *video player* dengan desain sendiri dapat memilih opsi *no skin*. Pada contoh di buku ini digunakan **skinOverAll.swf**. *File skin* tersebut nantinya akan muncul di *folder* tempat bekerja, dan diperlukan ketika proses *publishing*.



Gambar 192. pemilihan *skin*

Apabila proses impor video telah selesai, maka pada panel terakhir muncul tombol ***finish***. Tekan tombol *finish* tersebut dan video akan muncul beberapa saat kemudian di *Stage*.

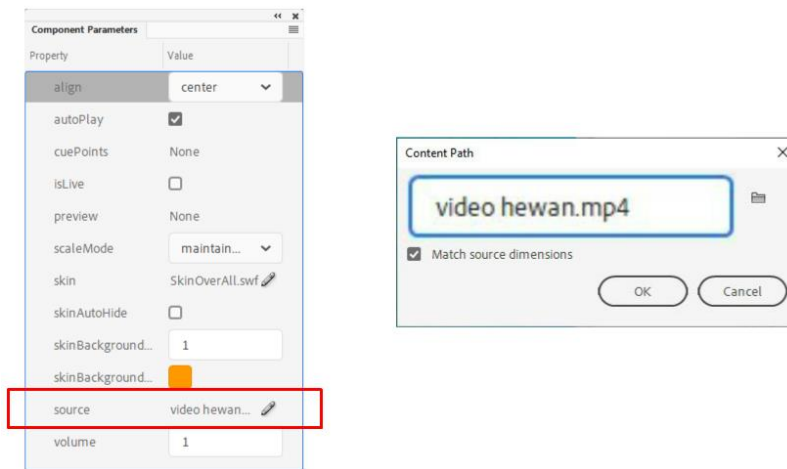


Gambar 193. tampilan setelah proses impor video

5. Dalam penggunaan video, terdapat beberapa hal yang harus diperhatikan agar video dapat dijalankan di berbagai perangkat nantinya. Sering kali video tidak dapat dimainkan ketika aplikasi dijalankan di perangkat lain, misalnya ketika dijalankan di komputer lainnya. Untuk mengatasi permasalahan umum tersebut, **klik video** kemudian buka panel ***Component Parameters*** (dengan menekan menu Window > Component Parameters)

Pada panel *Component Parameters*, terdapat beberapa pengaturan terkait video. *autoPlay* digunakan untuk menentukan apakah video langsung dimainkan ataukah

harus menunggu pengguna menekan tombol *play*. *Skin Background* dapat diinput dengan angka 1 untuk menghasilkan *skin* dengan warna yang *solid*. Dan yang terpenting adalah pada bagian **source**, klik pada bagian *source* untuk memastikan bahwa **content path** hanya berisi nama *file* saja dan tidak ada nama *folder*.



Gambar 194. pengaturan component parameter video

6. Setelah video ditambahkan, perlu ditambahkan **instance name**. Ketikkan “**videoku**” sebagai *instance name* video.
7. Untuk mengaktifkan halaman video, terlebih dahulu **klik Frame 10 Layer konten** kemudian tambahkan **instance name** “**btn_video**” pada tombol video.
8. Selanjutnya **klik Frame 10 Layer kode** dan tambahkan kode berikut :

```
mpi.click("btn_video", 70);
```

9. **Klik Frame 70 Layer kode** kemudian ketikkan kode berikut untuk mengaktifkan tombol *home* dan mengatur video:

```
mpi.click("btn_home", 10);  
mpi.video("videoku");
```

10. Jalankan aplikasi dan ujicoba halaman video.

Kode `mpi.video` secara otomatis akan menghentikan suara latar yang aktif agar tidak bertabrakan dengan suara video. Ketika pindah ke halaman lain, suara aktif akan dimainkan kembali secara otomatis dan video akan dihentikan secara otomatis.

Sampai dengan tahapan penambahan audio dan video, proyek multimedia interaktif “Tebak Satwa” sudah memiliki fitur-fitur yang cukup lengkap. Pada bab selanjutnya akan ditambahkan beberapa fitur pendukung yang akan meningkatkan interaktivitas aplikasi yang sedang dikembangkan.

BAB 8

Fitur Pelengkap Multimedia Interaktif

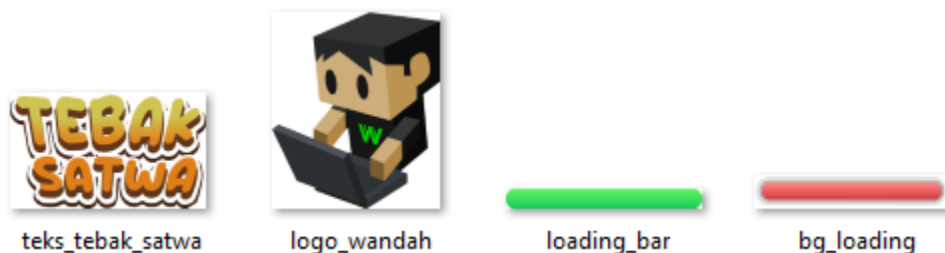
8.1 Intro

Intro merupakan sebuah tampilan pembuka dari sebuah aplikasi, dapat berupa *splash screen*, efek *loading* aplikasi, animasi dan lainnya. Intro dalam multimedia interaktif pada dasarnya merupakan fitur pelengkap yang sering ditemui, biasanya digunakan untuk menampilkan nama dan logo pengembang dari aplikasi tersebut. Intro dalam aplikasi multimedia interaktif dapat dibuat secara terpisah, dalam artian *file* intro merupakan *file* tersendiri yang selanjutnya dibuka oleh aplikasi, atau dibuat secara langsung di dalam aplikasi.

Pada contoh berikut akan dijelaskan proses pembuatan intro multimedia interaktif dalam format *file* terpisah. Perhatikan langkah-langkah berikut:

1. **Buatlah sebuah *file* baru** dengan pengaturan yang sama dengan proyek multimedia interaktif “Tebak Hewan” (1280x720 *pixel*, 30 *fps*). Simpan dengan nama **intro.fl**.
2. Pada *file* ini dapat dibuat berbagai macam konten, seperti *splash screen*, efek *loading* atau animasi. Teknik pembuatannya dapat memanfaatkan penggunaan *motion tween*, *motion guide*, atau sebatas tampilan statis dengan durasi *Frame* tertentu.

Pada contoh di buku ini, dibuat sebuah intro yang sederhana, yaitu efek *loading* dan *splash screen*. Terlebih dahulu disiapkan aset visual yang akan dibutuhkan untuk menyusun intro, yaitu latar untuk efek *loading*, *loading bar*, teks tebak satwa dan logo.



Gambar 195. persiapan aset visual untuk intro

3. Pada *file* FLA yang baru dibuat, **buatlah 3 buah *Layer***. Pada dasarnya penambahan *Layer* tidak ada ketentuan pasti, karena menyesuaikan kebutuhan masing-masing desain yang akan dibuat. Penambahan *Layer* dapat dilakukan pada tahapan berikutnya jika dibutuhkan, terutama ketika membuat animasi *motion* yang membutuhkan sebuah *Layer* (*Keyframe*) untuk sebuah objek yang dianimasikan.

Klik **Frame 1 Layer 1** pada posisi ini diletakkan gambar statik untuk *background*. **Buatlah sebuah kotak** berwarna untuk latar, lalu **impor file bg_loading** dan file **teks_tebak_satwa**.

Atur posisi masing-masing objek seperti pada gambar.



Gambar 196. posisi objek di *Frame 1 Layer 1*

Klik **Frame 1 Layer 2**, kemudian **impor file loader_bar**. Posisikan di atas gambar *bg_loading*. Seleksi gambar tersebut kemudian **convert** menjadi *MovieClip* "loading_bar".



Gambar 197. posisi *MovieClip* loading_bar

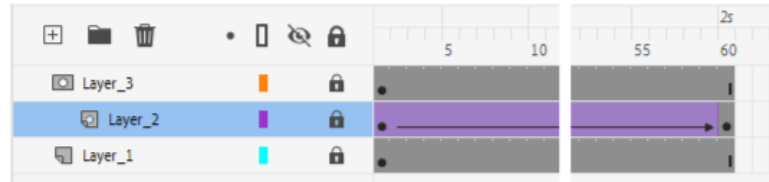
4. Klik **Frame 60 Layer 1** dan **Frame 60 Layer 3** tambahkan **Frame (F5)**. Selanjutnya klik **Frame 60 Layer 2**, tambahkan **Keyframe (F6)**. Pada *Layer 2* akan ditambahkan animasi *motion tween* agar *MovieClip* loading_bar menampilkan efek *loading* dari posisi sedikit sampai penuh.

Klik **Frame 1 Layer 2** kemudian dengan *transform tool*, geser posisinya ke sebelah kiri. Tambahkan **Classic Tween** untuk menghasilkan animasi gerak *MovieClip* loading bar dari kiri ke kanan.



Gambar 198. posisi *MovieClip* loading_bar pada *Frame 1*

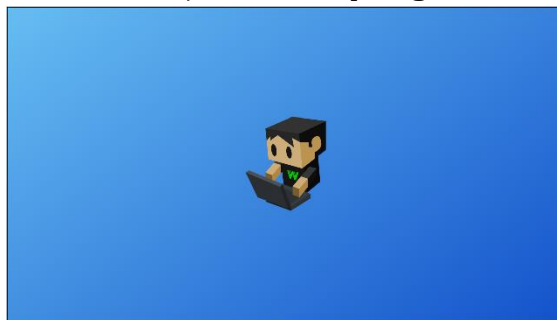
Untuk mendapatkan efek *loading* yang baik, klik **Frame 1 Layer 3** buatlah sebuah **rounded rectangle** dengan ukuran yang tepat dengan ukuran gambar **bg_loading**. Klik kanan *Layer 3* kemudian pilih opsi **Mask**.



Gambar 199. struktur *Layer*

5. Selanjutnya akan ditambahkan *splash screen* berupa logo dengan efek *fade in fade out*. Klik **Frame 61 seluruh *Layer*** kemudian tambahkan ***Blank Keyframe***.

Klik **Frame 61 *Layer 1***, dan buatlah kotak untuk latar belakang. Selanjutnya, klik **Frame 61 *Layer 2*** kemudian **import** gambar logo ke *Stage*, letakkan tepat di tengah. **Convert** gambar logo tersebut menjadi ***MovieClip "logo_mc"***.

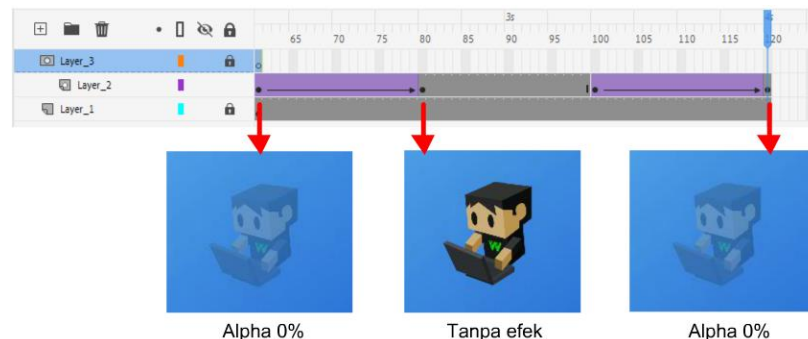


Gambar 200. struktur pada *Frame 61* dan penempatan logo

6. Klik **Frame 120 *Layer 1***, tambahkan ***Frame (F5)***. Selanjutnya untuk membuat animasi *fade in fade out* pada logo dibutuhkan 4 buah ***Keyframe***. Klik ***Layer 2***, tambahkan ***Keyframe (F6)*** pada ***Frame 80, Frame 100*** dan ***Frame 120***.

Pada ***Frame 61*** dan ***Frame 120***, klik ***MovieClip "logo_mc"***, buka panel ***Properties*** dan tambahkan efek ***Alpha 0%*** untuk menghasilkan efek transparan.

Seleksi ***Frame 61 sampai Frame 120***, kemudian tambahkan ***Classic Tween***, untuk menghasilkan animasi dengan efek *fade in dan fade out*.



Gambar 201. struktur *Timeline* efek *fade in fade out* logo

7. Simpan, dan jalankan *file* dengan menekan **Ctrl+Enter**, maka akan terbentuk *file intro.swf*. Pastikan *file* intro berada satu *folder* dengan proyek multimedia interaktif “Tebak Satwa”.

Setelah intro selesai dibuat, intro akan ditambahkan ke dalam aplikasi dengan meletakkannya pada *Frame 2*. Apabila muncul pertanyaan, kenapa *file* intro tidak diletakkan di *Frame 1*?, jawaban atas pertanyaan tersebut adalah dalam beberapa versi Adobe Animate memiliki perilaku yang berbeda terhadap proses membuka (*loading*) sebuah *file* atau objek. Simbol MPI memiliki kode yang terkompilasi dalam jumlah besar dan memiliki ukuran *file* 3MB. Untuk membuka *file* tersebut diperlukan waktu sepersekian detik. Untuk membuka sebuah intro akan digunakan kode mpi, sehingga apabila kode diletakkan di *Frame 1* dan *file* mpi belum sepenuhnya terbuka maka akan muncul *error*. Sementara, apabila diletakkan pada *Frame 2*, mpi sudah dipastikan telah terbuka seutuhnya sehingga kode akan terdeteksi.

Untuk menambahkan intro ke dalam proyek multimedia interaktif “Tebak Satwa”, ikuti langkah-langkah berikut :

1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**).
2. Klik **Frame 2 Layer kode** tambahkan **Blank Keyframe**.
3. Buka panel **Action** dan tambahkan kode berikut :

```
stop();  
mpi.bukaSWF("intro.swf", mulai);  
  
function mulai() {  
    gotoAndStop(10);  
}
```

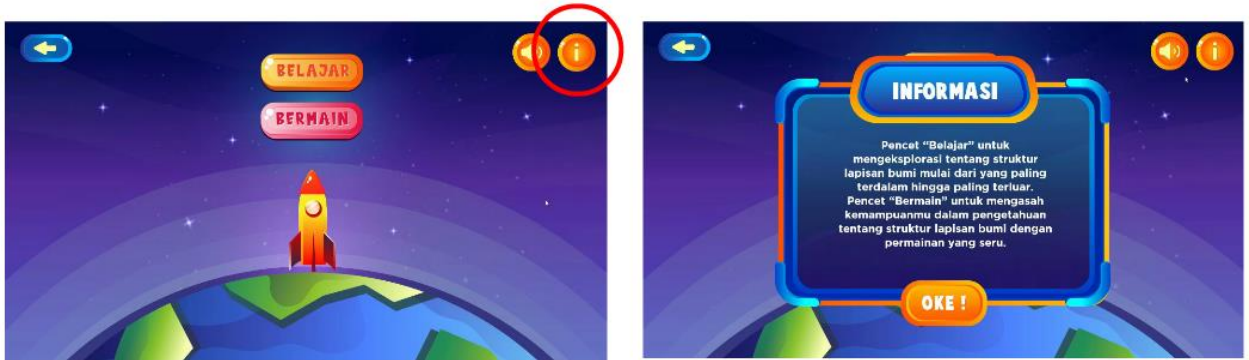
4. Jalankan aplikasi untuk menguji fitur intro.

Kode `mpi.bukaSWF` akan membuka *file* “`intro.swf`” dan akan menjalankan fungsi `mulai` setelah animasi pada *file* intro selesai. Pada fungsi `mulai`, aplikasi akan diarahkan ke *Frame* 10 tempat halaman judul berada.

8.2 Popup

Popup adalah sebuah area tampilan antarmuka yang dapat muncul sebagai jendela informasi dalam berbagai ukuran dan posisi. *Popup* dapat digunakan untuk memberitahukan sebuah informasi tertentu kepada pengguna, sebagai contoh untuk menampilkan peringatan, petunjuk/bantuan, atau sekadar menyampaikan informasi terkait fitur tertentu dalam sebuah aplikasi. Sebagai contoh multimedia interaktif “Menenal Lapisan Bumi” karya Siti F.B. Oktafiardiani (2022) pada setiap halamannya menyediakan tombol informasi, yang ketika

ditekan akan memunculkan *popup* informasi. Hal tersebut akan membantu pengguna dalam memahami langkah-langkah apa saja yang dapat dilakukan di halaman tersebut.



Gambar 202. contoh *Popup* dalam multimedia interaktif “Mengenal Lapisan Bumi”
(Oktafiardiani, 2022)

Untuk menambahkan *popup* ke dalam aplikasi multimedia interaktif, perhatikan langkah-langkah berikut :

1. Buka kembali file (**tutorial 8 – Tebak Satwa**) dan file MPI-component.flr.
2. Tombol *popup* akan diletakkan di halaman di halaman judul dan apabila diklik akan menampilkan informasi terkait aplikasi.

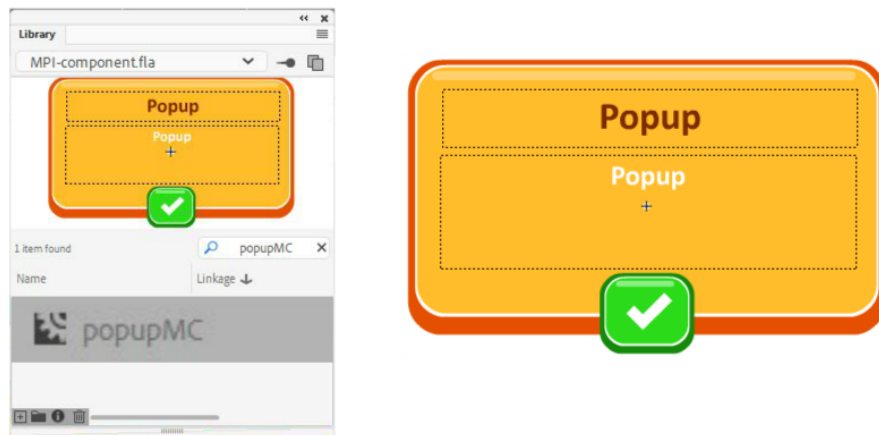
Klik **Frame 10 Layer konten**, kemudian buatlah sebuah tombol “**btn_info**”, tambahkan *instance name* “**btn_info**”.



Gambar 203. menambahkan tombol “btn_info”

3. Untuk menambahkan *popup*, buka **Library** file **MPI-component**, lalu drag **MovieClip popupMC** dari **Library** ke **Stage**. Selanjutnya hapus **MovieClip** **popupMC** dari **Stage**. Jika diperhatikan **MovieClip** **popupMC** sudah memiliki **Linkage**, sehingga dapat ditambahkan ke layar menggunakan kode.

MovieClip popupMC dapat di edit sesuai kebutuhan, di dalamnya terdapat 2 buah *dynamic text* yang secara dinamis dapat diatur dengan kode serta sebuah tombol yang digunakan untuk menutup *popup*.



Gambar 204. *MovieClip* popupMC

4. Untuk mengaktifkan fitur *popup*, perlu ditambahkan sedikit kode. **Klik *Frame 10 Layer code***, kemudian tambahkan kode berikut :

```
mpi.click("btn_info", tampilkanPopup);

function tampilkanPopup():void{
    mpi.popup("popupMC", "Tebak Satwa", "Selamat datang di
    aplikasi tebak satwa, klik salah satu tombol untuk memulai");
}
```

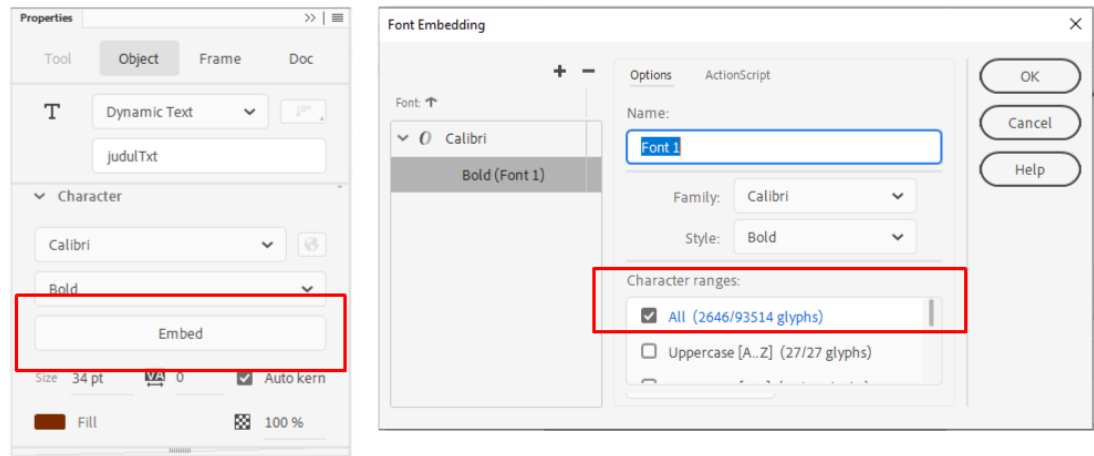
5. Jalankan aplikasi untuk mengujicoba fitur *popup*. Maka akan muncul pesan di panel ***Output***, yaitu :

Fonts should be *Embedded* for any text that may be edited at runtime, other than text with the "Use Device Fonts" setting. Use the Text > Font *Embedding* command to *Embed* fonts.

Hal ini berdampak pada tampilan *popup*, huruf tidak akan ditampilkan dengan benar. Di dalam aplikasi Adobe Animate, sebuah *dynamic text* membutuhkan *file* huruf (**Font**) agar tampil dengan benar. Oleh karena itu *file* huruf perlu disertakan ke dalam aplikasi atau diistilahkan dengan ***Embed Font***.

Untuk mengatasi permasalahan tersebut, buka ***Library*** dan **dobel klik *MovieClip* "popupMC"** untuk mengeditnya. Dalam mode edit, **klik *dynamic text* "Judul Popup"**, buka panel ***Properties*** dan klik tombol ***Embed***. Pilih opsi ***All*** dan klik ***OK***, maka *file* huruf akan disertakan ke dalam aplikasi. Penambahan *file* huruf tidak berdampak

signifikan terhadap *file* hasil akhir, sebuah set huruf rata-rata hanya menambah ukuran *file* 50-300 Kilobyte.



Gambar 205. menambahkan *file* huruf (*Embed font*)

6. Jalankan kembali aplikasi, dan ujicoba fitur *popup*. Dengan menambahkan *file* huruf, maka *popup* dapat menampilkan teks dengan benar.



Gambar 206. hasil penambahan fitur *popup*

Fungsi `mpi.popup("Linkage", "judul", "pesan")` memiliki beberapa parameter, yaitu `"popupMC"`, yang merupakan *Linkage* dari *MovieClip* yang akan ditambahkan. Parameter lain adalah `"Judul popup"` dan `"pesan popup"`. Variasi lain dari `mpi.popup` adalah :

```
//jika ingin menjalankan fungsi ketika popup ditutup
mpi.popup("popupMC", "Judul Popup", "Isi pesan popup", fungsiPop);
//jika ingin menampilkan popup pada posisi tertentu x, y
mpi.popup("popupMC", "Judul Popup", "Isi pesan popup", null,
200,300);
```

Apabila ingin menambahkan *instance name* baru di dalam *MovieClip* popupMC dan ingin mengakses *instance name* tersebut, maka dapat dilakukan pengaturan *popup* sebagai variabel. Sebagai contoh ditambahkan sebuah input text dengan *instance name* nama_txt, maka kode untuk mengaksesnya adalah :

```
var popup = mpi.popup("popupMC", "Selamat Datang", "Perkenalkan
dirimu", sambutan);

function sambutan(){
    var nama = popup.nama_txt.text;
    trace("Haloo "+nama);
}
```

8.3 Login

Dalam beberapa aplikasi mengharuskan pengguna untuk melakukan proses *login* terlebih dahulu agar data pengguna dapat direkam oleh aplikasi. Membangun sistem registrasi dan sistem *login* suatu aplikasi adalah sebuah proses yang kompleks karena melibatkan penyimpanan *database* ke *server*. Data pengguna sekaligus keamanan pengguna menjadi salah satu tantangan dalam pengembangan sistem *login*.

Terkait dengan sistem *login*, *MPI Component* menyediakan sistem *login* yang lebih sederhana, sekadar menyimpan data pengguna secara lokal (hanya terkait dengan perangkat), tidak melibatkan *server*. Setidaknya sistem *login* yang ditawarkan mudah untuk dioperasikan dan dapat digunakan untuk keperluan tertentu seperti menyimpan score atau menyimpan capaian (*progress*) suatu materi tertentu.

Untuk menambahkan sistem *login* pada proyek multimedia interaktif “Tebak Satwa” ikuti langkah-langkah berikut:

1. Buka kembali *file* (tutorial 8 – Tebak Satwa) dan *file* MPI-component fla.
2. Tombol *login* akan diletakkan di halaman di halaman judul di samping tombol pengaturan suara. Klik **Frame 10 Layer** konten. Buka **Library** *file* MPI-Component, kemudian **drag MovieClip “loginMC”** ke *Stage*. Tambahkan ***instance name* “loginMC”** dan letakkan di pojok kiri atas.



Gambar 207. menambahkan *MovieClip* loginMC

3. Untuk mengaktivasi fitur *login*, klik **Frame 10 Layer kode**, kemudian tambahkan kode berikut :

```
mpi.login("loginMC");
```
4. Jalankan aplikasi dan ujicoba sistem *login*. Tutup panel swf, kemudian jalankan kembali aplikasi, maka nama yang diinput akan terekam oleh sistem.

Kode `mpi.login` akan merekam nama pengguna dan menyimpannya secara lokal ke dalam perangkat. Ketika aplikasi dimatikan atau perangkat dimatikan, data nama yang terekam akan tetap ada. Untuk mengakses nama pengguna yang telah diinputkan dapat digunakan kode `mpi.nama`.

8.4 Drag and drop

Drag and drop adalah istilah yang digunakan pada kegiatan menyeleksi sebuah objek virtual lalu menyeretnya ke posisi tertentu. Sebagai contoh dalam multimedia interaktif “Bersih-bersih” karya Marchya Lidya (2021) terdapat sebuah halaman yang mengajak pengguna untuk membersihkan suatu ruangan dengan teknik *drag and drop*.

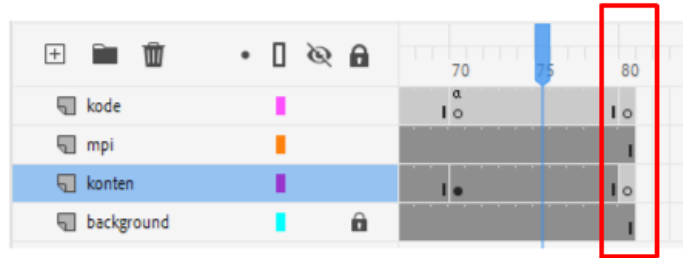


Gambar 208. fitur *drag and drop* pada multimedia interaktif “Bersih-bersih” (Lidya, 2021)

8.4.1 Menambahkan Halaman Permainan

Pada multimedia interaktif “Tebak Satwa” fitur *drag and drop* akan diletakkan pada halaman permainan. Pada halaman permainan akan dibagi menjadi 3 jenis yaitu *drag and drop*, kuis dan *puzzle*. Perhatikan langkah-langkah berikut untuk menambahkan halaman permainan:

1. Buka kembali file (**tutorial 8 – Tebak Satwa**) dan file *MPI-component fla*.
2. Halaman permainan akan diletakkan di *Frame 80*, oleh karena itu tambahkan **Blank Keyframe** pada *Frame 80 Layer kode* dan *Layer konten*, dan tambahkan **Frame** pada *Layer mpi* dan *Layer background*.



Gambar 209. struktur *Timeline* pada *Frame* 80

3. **Klik *Frame* 80 *Layer* konten**, kemudian buatlah konten untuk halaman permainan yang terdiri dari 4 buah tombol yaitu “btn_home”, “btn_permainan1”, “btn_permainan2” dan “btn_permainan3”. Tambahkan juga ***instance name*** sesuai nama masing-masing tombol.



Gambar 210. struktur halaman permainan

4. Untuk mengaktifkan halaman permainan perlu ditambahkan kode pada halaman judul dan halaman permainan. **Klik *Frame* 10 *Layer* konten**, pastikan tombol permainan memiliki ***instance name*** “btn_permainan”, selanjutnya **klik *Frame* 10 *Layer* kode** kemudian tambahkan kode berikut :

```
mpi.click("btn_permainan", 80);
```

Sedangkan pada halaman permainan perlu ditambahkan kode untuk mengaktifkan 4 buah tombol yang ada. Khusus untuk halaman permainan 1, 2 dan 3 direncanakan pada *Frame* 90, *Frame* 100 dan *Frame* 110. Untuk mengaktifkan tombol pada halaman permainan, **klik *Frame* 80 *Layer* kode**, kemudian ketikkan kode berikut:


```

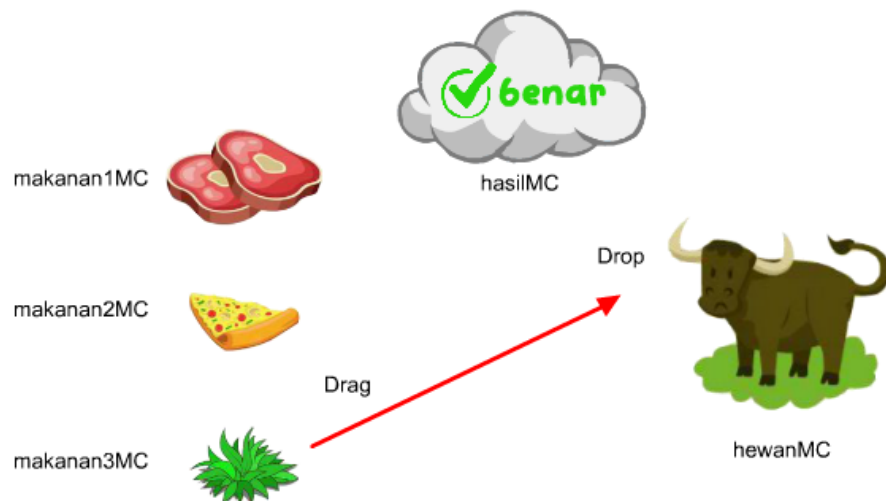
mpi.click("btn_home", 10);
mpi.click("btn_permainan1", 90);
mpi.click("btn_permainan2", 100);
mpi.click("btn_permainan3", 110);

```

5. Jalankan aplikasi dan ujicoba untuk menuju ke halaman permainan. Pada tahapan ini tombol halaman permainan belum dapat diakses karena belum ada *Frame* yang dituju.

8.4.2 Permainan *Drag and drop*

Setelah halaman permainan dibuat, maka tahapan berikutnya adalah menambahkan fitur permainan *drag and drop* pada *Frame* 90. Konsep permainan *drag and drop* yang akan dibuat adalah mencocokkan makanan hewan.



Gambar 211. konsep permainan *drag and drop*

Pada konsep di atas, terdapat 3 buah *MovieClip* makananMC, yang masing-masing dapat di *drag* dan di *drop* ke *MovieClip* hewanMC. Apabila makanan di *drop* tepat di hewan, maka akan ditampilkan *MovieClip* hasilMC (yang berisi benar dan salah). Permainan akan di ulangi sebanyak 5 kali dengan sistem pengacakan.

Untuk membuat permainan *drag and drop* tersebut, perlu disiapkan beberapa aset visual seperti gambar makanan, gambar hewan dan gambar untuk *MovieClip* hasil. Pada tutorial ini digunakan 4 gambar hewan karnivora, 4 gambar herbivora, 4 gambar makanan dan gambar untuk benar dan salah.

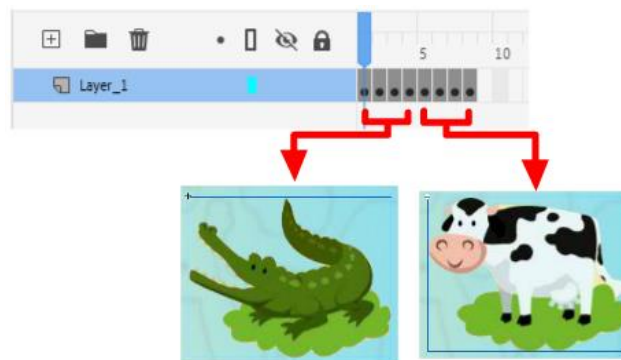


Gambar 212. persiapan aset visual

Perhatikan langkah berikut untuk membuat permainan dengan fitur *drag and drop* :

1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**) dan *file* MPI-component fla.
2. **Klik Frame 90** tambahkan **Blank Keyframe** pada **Layer kode** dan **Layer konten**, dan tambahkan **Frame** pada **Layer mpi** dan **Layer background**.
3. **Klik Frame 90 Layer konten**, untuk membuat *MovieClip* hewanMC, **import gambar “hewan_karnivora1”** ke *Stage*, apabila ada panel untuk mengimpor gambar secara *sequence* (atau diimport semua gambar yang berurutan), pilih opsi tidak (hanya dibutuhkan 1 gambar terlebih dahulu). **Convert** gambar tersebut menjadi *MovieClip* “hewanMC” dengan titik registrasi di pojok kiri atas.

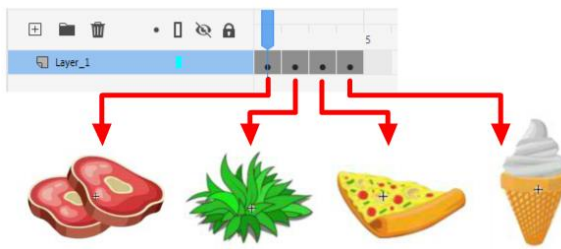
Dobel klik *MovieClip* “hewanMC” untuk mengeditnya. Pada mode edit, **klik Frame 2** tambahkan **Blank Keyframe**, kemudian **import gambar “hewan_karnivora2”** secara *sequence*, maka akan terbentuk susunan hewan secara *Frame by Frame*. **Import** juga gambar *file* herbivora secara *Frame by Frame*, sehingga dalam *MovieClip* “hewanMC” terdapat beberapa gambar hewan yang tertata secara *Frame by Frame*. Dalam contoh di buku ini, *Frame* 1-4 adalah hewan karnivora dan *Frame* 5-8 adalah hewan herbivora.



Gambar 213. struktur *MovieClip* hewanMC

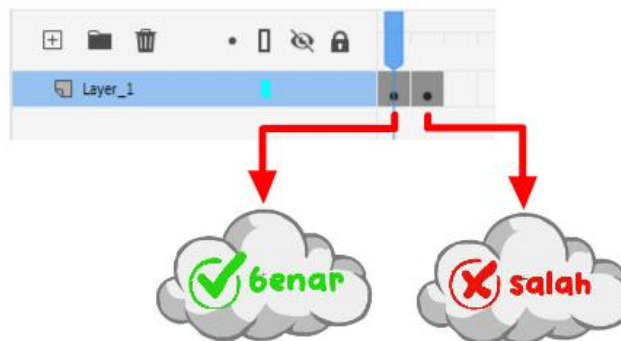
Keluar dari mode edit *symbol* dan tambahkan **instance name “hewanMC”** pada *MovieClip* tersebut.

4. **Import file** gambar **makanan_daging**, kemudian **convert** menjadi **MovieClip** “**makananMC**”. Edit **MovieClip** tersebut, dan **tambahkan gambar** rumput (**Frame 2**), dan makanan lainnya (**Frame 3-4**). Perhatikan gambar :



Gambar 214. struktur *MovieClip* makananMC

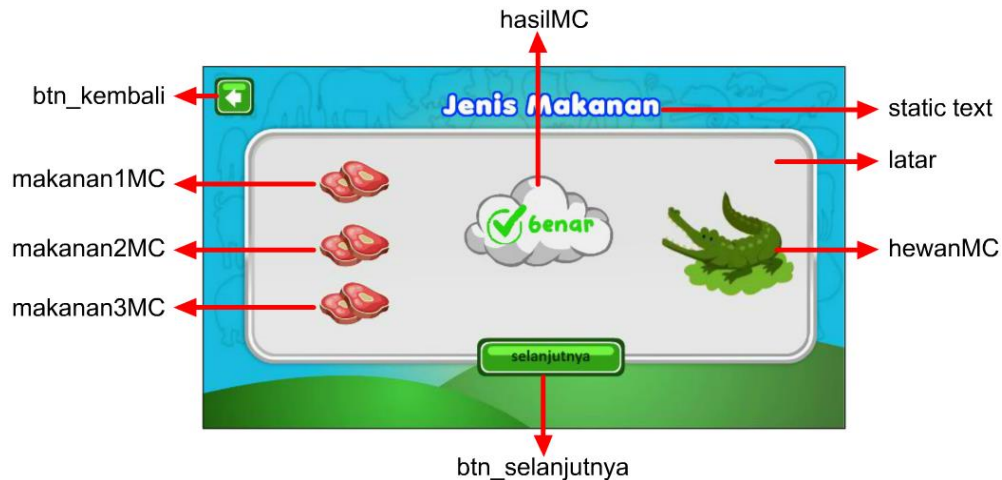
5. Keluar dari mode edit simbol. **Copy-paste** **MovieClip** makananMC sebanyak 2 kali (sehingga terdapat 3 buah **MovieClip** makananMC di layar). Tambahkan **instance name** “**makanan1MC**”, “**makanan2MC**”, “**makanan3MC**”.
6. Sebagai indikator benar dan salah, akan dibuat sebuah **MovieClip** hasilMC. **Import file** gambar awan dan icon benar. Tambahkan **Static Text** “**benar**”, untuk mempercantik tampilan. Kemudian **convert** menjadi **MovieClip** “**hasilMC**”. Edit **MovieClip** hasilMC, pada **Frame 2** tambahkan **Blank Keyframe** dan buatlah tampilan untuk menampilkan hasil salah.



Gambar 215. struktur *MovieClip* hasilMC

Keluar dari mode edit *symbol* dan tambahkan **instance name** “**hasilMC**”.

7. Tambahkan beberapa elemen lain seperti tombol “**btn_kembali**”, tombol “**btn_selanjutnya**”, dan latar belakang untuk pemanis tampilan. Atur posisi masing-masing elemen pada *Stage* dan pastikan masing-masing objek memiliki **instance name**.



Gambar 216. objek pada *Frame 90 Layer* konten

8. Klik **Frame 90 Layer kode**, kemudian ketikkan kode berikut untuk mengaktifkan fitur *drag and drop*:

```

mpi.click("btn_kembali", 80);
mpi.click("btn_selanjutnya", naik_level);

//mulai permainan
setup_gamel();
//tampilkan popup bantuan
mpi.popup("popupMC", "Makanan Hewan", "Drag makanan hewan yang tepat!");

mpi.level_game = 1; //mengatur level 1
mpi.level_maks = 5; //level maksimum
mpi.nilai_game = 0; //nilai permainan

//fungsi untuk mengatur level
function setup_gamel(){
    //acak hewan
    hewanMC.gotoAndStop(mpi.acak(8));
    //mengatur drag makananMC
    mpi.resetMC("makanan1MC");
    mpi.resetMC("makanan2MC");
    mpi.resetMC("makanan3MC");
    mpi.drag("makanan1MC", cekMakanan);
    mpi.drag("makanan2MC", cekMakanan);
    mpi.drag("makanan3MC", cekMakanan);
    makanan1MC.gotoAndStop(1);
    makanan2MC.gotoAndStop(2);
    makanan3MC.gotoAndStop(mpi.acak(4)+1);
    //sembunyikan hasil
    hasilMC.visible = false;
    btn_selanjutnya.visible = false;
}

```

```

//fungsi dijalankan setelah drag selesai
function cekMakanan() {
    var makanan = mpi.dragOb;
    //cek ketika makanan menyentuh hewan
    if (makanan.hitTestObject(hewanMC)) {
        //matikan drag
        makanan1MC.drag = false;
        makanan2MC.drag = false;
        makanan3MC.drag = false;
        //cek benar atau salah
        if (makanan.currentFrame == 1 && hewanMC.currentFrame<=4) {
            //karnivora
            hasilMC.gotoAndStop(1);
            mpi.nilai_game+=20;
        }else if (makanan.currentFrame == 2 &&
hewanMC.currentFrame>4) {
            //herbivora
            hasilMC.gotoAndStop(1);
            mpi.nilai_game+=20;
        }else{
            //salah
            hasilMC.gotoAndStop(2);
        }
        //tampilkan hasil
        hasilMC.visible = true;
        btn_selanjutnya.visible = true;
    }else{
        //jika tidak menyentuh hewan
        //MovieClip akan direset posisinya
        mpi.resetMC(makanan);
    }
}

function naik_level() {
    mpi.level_game++;
    //permainan selesai
    if (mpi.level_game>mpi.level_maks) {
        //tampilkan nilai dalam popup
        mpi.popup("popupMC", "Permainan Selesai", "Selamat
"+mpi.nama+" Nilai mu "+mpi.nilai_game, tutup_permainan1);
    }else{
        //ulangi permainan
        setup_gamel();
    }
}

function tutup_permainan1() {
    gotoAndStop(80);
}

```

9. Jalankan aplikasi dan ujicoba permainan.

Kode `mpi.drag("makananMC")` secara otomatis akan menambahkan fitur *drag* kepada *MovieClip* `makananMC`. Apabila ditambahkan parameter fungsi seperti `mpi.drag("makananMC", cekMakanan)` maka fungsi `cekMakanan` akan dijalankan setelah *mouse* dilepas (*drop*).

8.5 Kuis

Kuis ialah suatu format evaluasi dalam bentuk soal ataupun pertanyaan yang memungkinkan pengguna untuk meningkatkan wawasan dan pengetahuannya secara mandiri. Pemberian kuis dalam aplikasi multimedia merupakan salah satu cara untuk melatih pengguna aplikasi melakukan refleksi terhadap materi yang diajarkan, sehingga dapat membantu membangun kemampuan melakukan asesmen diri atas tingkat pemahaman yang dicapai.

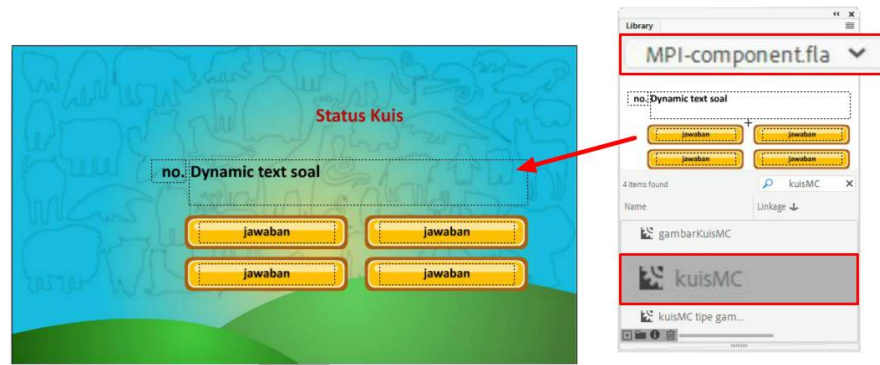
Dalam *file* *MPI-component* terdapat 3 format kuis yang dapat digunakan :

1. Kuis pilihan ganda, dengan jumlah jawaban 2 sampai 5.
2. Kuis pilihan ganda dengan dukungan pertanyaan berupa gambar.
3. Kuis pilihan ganda dengan jawaban berupa gambar.

Pada contoh di buku ini akan digunakan format kuis pilihan ganda dengan jawaban berupa teks dan memiliki dukungan soal berupa gambar. Kuis akan diletakkan pada *Frame* 100, sesuai dengan pengaturan sebelumnya. Perhatikan langkah-langkah berikut untuk menambahkan fitur kuis :

1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**) dan *file* *MPI-component fla*.
2. **Klik *Frame 100* tambahkan *Blank Keyframe* pada *Layer kode* dan *Layer konten*, dan tambahkan *Frame* pada *Layer mpi* dan *Layer background*.**
3. Untuk menambahkan fitur kuis, dapat mengambil secara langsung *MovieClip* kuis dari *Library file MPI Component*. **Klik *Frame 100 Layer konten*, buka panel *Library file MPI-Component*, kemudian *drag MovieClip kuisMC* ke *Stage*. Tambahkan *instance name* "**kuisMC**"**

MovieClip `kuisMC` memiliki beberapa kelengkapan untuk menampilkan fitur kuis seperti *dynamic text*, 4 tombol jawaban (dapat dikurangi atau diperbanyak dengan cara *copy-paste* dan memastikan *instance name* nya sesuai), serta *MovieClip* status kuis untuk menunjukkan status jawaban pemain. Grafis *MovieClip* ini dapat diedit sesuai dengan kebutuhan, yang terpenting adalah tidak merubah struktur *Timeline* dan penamaan *instance name*. Khusus untuk *dynamic text*, apabila menggunakan jenis huruf yang berbeda, pastikan untuk mengaktifkan fitur ***Embed***.



Gambar 217. menambahkan kuisMC

4. Untuk menambahkan gambar, sebagai penunjang soal dalam kuis, maka disiapkan gambar hewan dengan nama *file* yang berurutan. Pada contoh ini digunakan *file* hewan1 – hewan16.



Gambar 218. *file* hewan yang akan diimpor sebagai gambar soal kuis

Import satu buah gambar hewan1.png, kemudian **convert** menjadi **MovieClip** “gambarKuisMC”. **Dobel klik** **MovieClip** tersebut untuk mengeditnya, kemudian tambahkan **Blank Keyframe** pada **Frame 2** dan **import** gambar hewan berikutnya secara *Frame by Frame (sequence)*. Dengan teknik ini **MovieClip** gambarKuisMC akan berisi 16 *Frame* berisi gambar hewan yang berbeda-beda.

Keluar dari mode edit *symbol* dan tambahkan **instance name** “gambarKuisMC”.

5. Untuk kelengkapan halaman, tambahkan tombol “**btn_kembali**” pada pojok kiri atas, dan tambahkan latar agar kuis dapat tampil dengan baik.



Gambar 219. stuktur halaman kuis

8.5.1 Membuat *database* soal untuk kuis

Dalam pembuatan kuis, hal spesifik yang perlu diperhatikan adalah soal kuis yang akan digunakan. Seperti halnya pada tutorial di buku sebelumnya, soal kuis akan diletakkan dalam variabel bertipe *Array*. *Array* adalah kumpulan elemen dengan tipe yang sama yang ditempatkan di lokasi memori yang berdekatan yang dapat direferensikan secara individual dengan menggunakan indeks ke pengidentifikasi unik. Perhatikan contoh *array* sederhana berikut :

```
var hari = ["Minggu", "Senin", "Selasa", "Rabu", "Kamis", "Jumat", "Sabtu"];
trace( hari[1]); //Senin
```

Pada contoh di atas, data nama-nama hari disimpan dalam sebuah variabel *hari*, untuk mengakses data yang ada pada baris ke dua dituliskan *hari[1]*, yang berarti mengakses data urutan ke 1, yaitu “Senin”, karena perhitungan data dimulai dari angka 0.

Pada pembuatan kuis, soal juga dimasukkan ke dalam format *array* untuk mempermudah proses pengacakan, dan pengecekan jawaban. Perhatikan struktur variabel soal yang digunakan oleh sistem kuis berikut :

```
var soal = [{"Hewan apakah itu?", "Rubah", "Tikus", "Tupai", "Binturong", 2},
            ...
            ]
```

Pada contoh di atas, variabel *soal* memiliki elemen yang bertipe *array*, pada dunia pemrograman disebut sebagai *array* bertingkat. Untuk mengakses soal dengan nomor tertentu, maka dapat dituliskan kode *soal[0]* yang akan menghasilkan *array* ["Hewan apakah itu?", "Rubah", "Tikus", "Tupai", "Binturong", 2]. Pada *soal[0]* tersebut, elemen pertama (*soal[0][0]*) adalah soal yang akan muncul pada *dynamic text* (“hewan apakah itu?”), elemen kedua (*soal[0][1]*) adalah jawaban yang benar (“rubah”), 3 nama hewan berikutnya adalah jawaban pengecoh, sedangkan angka 2 adalah *Frame* tempat *MovieClip* gambarKuisMC (lihat gambar di atas, gambar nomor 2 adalah gambar rubah). Apabila soal tidak membutuhkan gambar, maka elemen terakhir dari *array* *soal[0]* dapat dihilangkan.

Variabel soal diletakkan di *Frame 100 Layer* kode, dengan jumlah soal sebisa mungkin 2 kali lipat dari jumlah soal yang akan dikerjakan oleh pengguna agar pengacakan soal berjalan dengan baik. Perhatikan contoh variabel soal berikut :

```
var soal = [{"Hewan apakah itu?", "Rubah", "Tikus", "Tupai", "Binturong",
2},
            [{"Hewan tersebut termasuk hewan?", "Omnivora", "herbivora",
"karnivora", "insectivora", 4},
            [{"Hewan darat apakah yang terbesar di dunia?", "Gajah", "ikan
Paus", "Badak", "Jerapah", 5},
```

```

        ["Hewan apakah yang memiliki duri pelindung?", "Landak",
"Durian", "Tupai", "Rubah", 3],
        ["Hewan pemakan tumbuhan disebut?", "Herbivora", "karnivora",
"omnivora", "Insectivora"],
        ["Hewan pemakan daging disebut?", "karnivora", "Herbivora",
"omnivora", "Insectivora"],
        ["Hewan apakah yang gemar memakan bambu?", "Panda", "Kuda",
"Tapir", "Kijang", 13],
        ...
        ["Hewan apakah yang paling tinggi?", "Jerapah", "Ilama",
"Burung", "Beruang", 16]];

```

Perhatikan pada baris terakhir dari variabel soal diakhiri dengan tanda `]]`; hal ini menunjukkan akhir dari variabel soal. Pada baris terakhir tersebut *programmer* sering kali mengalami kesalahan dalam menuliskan variabel *array* bertingkat dengan munculnya karakter koma (`,`).

8.5.2 Perprograman kuis

Setelah memahami struktur variabel soal, pada tahapan selanjutnya adalah menempatkan kode soal tersebut ke *Frame 100 Layer* kode. Perhatikan langkah berikut, yang merupakan lanjutan langkah-langkah sebelumnya :

6. **Klik *Frame 100 Layer* kode**, kemudian ketikkan kode berikut :

```

mpi.click("btn_kembali", 80);

var soal = [{"Hewan apakah itu?", "Rubah", "Tikus", "Tupai",
"Binturong", 2},
        ....
        ["Hewan apakah yang paling tinggi?", "Jerapah",
"Ilama", "Burung", "Beruang", 16] ];

mpi.kuis("kuisMC", soal, 5, selesaiKuis);
kuisMC.setGambar("gambarKuisMC");

function selesaiKuis() {
    mpi.popup("popupMC", "Kuis Selesai", "Selamat "+mpi.nama+"
nilai anda "+kuisMC.score,kembaliKuis);
}

function kembaliKuis() {
    gotoAndStop(80);
}

```

7. Jalankan aplikasi dan ujicoba fitur kuis.



Gambar 220. hasil penambahan fitur kuis

8.5.3 Penjelasan Program

Pada kode di atas, `mpi.kuis("kuisMC", soal, 5, selesaiKuis)` berfungsi untuk memulai kuis. Kuis akan ditampilkan pada *MovieClip* "kuisMC", menggunakan *database* dari variabel `soal`, soal maksimal sejumlah 5 dan akan memanggil fungsi `selesaiKuis` apabila pengguna sudah menjawab semua soal.

Secara umum jawaban berjumlah 4, namun apabila menginginkan jumlah jawaban lain dapat ditambahkan kode :

```
kuisMC.jumlahJawaban = 3;
```

Selain itu apabila menginginkan kuis dengan jawaban berupa gambar, dapat mengacu pada *file* MPI-component *Frame* 116 yang memberikan contoh kuis dengan jawaban berupa gambar.

Untuk menentukan format opsi jawaban dapat digunakan kode berikut:

```
kuisMC.opsi = "A"; //"", "A", "a", "1"
```

Fitur lain dari kuis adalah penggunaan suara. Suara dapat diimpor ke dalam *Library* dan ditambahkan *Linkage*. Selanjutnya penambahan suara dapat dilakukan dengan kode berikut :

```
kuisMC.suaraSoal = ""; //isi dengan Linkage suara
kuisMC.suaraJawab = "sound2"; //isi dengan Linkage suara
kuisMC.suaraBenar = "sound1,sound2"; //isi dengan Linkage suara
kuisMC.suaraSalah = "insound2,insound3 "; //isi dengan Linkage suara
kuisMC.suaraWaktuHabis = "soundTimeout"; //isi dengan Linkage suara
kuisMC.suaraSoalHabis = "soundOver"; //isi dengan Linkage suara
```


Apabila suara kosong, maka tidak akan memunculkan suara. Sedangkan apabila suara diisi dengan beberapa *Linkage*, maka suara akan dimunculkan secara acak. Sebagai contoh pada kode di atas, suara ketika jawaban salah diisi 3 buah *Linkage*, maka ketika pemain menjawab salah, salah satu dari suara tersebut akan dipilih secara acak dan dimainkan.

8.6 Puzzle

Permainan berikutnya yang akan ditambahkan ke dalam aplikasi adalah *puzzle*. *Puzzle* yang dimaksud adalah sebuah gambar yang dipecah menjadi beberapa kepingan, diacak dan meminta pemain untuk menyusun ulang. Permainan *puzzle* menuntut pemain untuk mengidentifikasi objek, sehingga mampu meningkatkan fokus pemain.

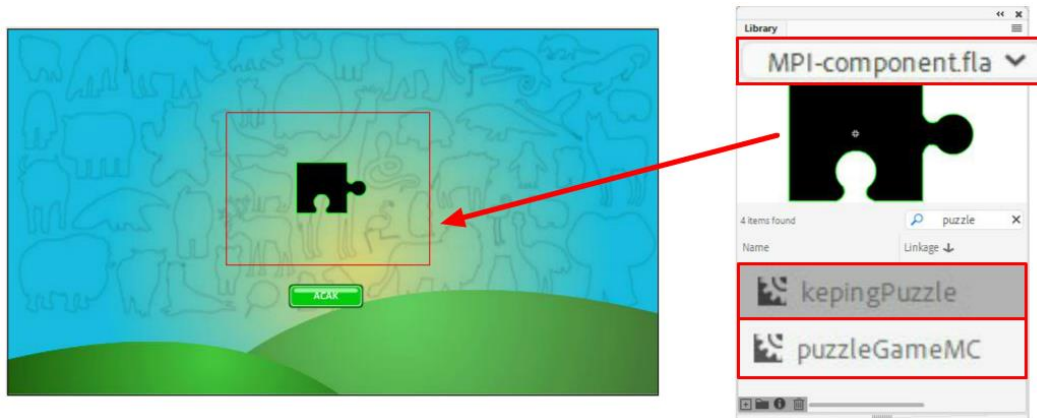
Untuk menambahkan *puzzle* terlebih dahulu disiapkan sebuah gambar untuk *puzzle*. Gambar yang digunakan untuk *puzzle* harus memiliki ukuran yang tepat, dengan kelipatan sesuai dengan ukuran kepingan *puzzle*. Secara default ukuran kepingan *puzzle* pada MPI-component adalah 100x100 *pixel*. Ukuran tersebut dapat disesuaikan nantinya dengan penambahan sedikit kode. Pada contoh ini digunakan ukuran yang tetap yaitu 100x100 *pixel* dengan jumlah kepingan 7x4 keping sehingga ukuran gambar yang disiapkan adalah 700x400 *pixel*.



Gambar 221. *file* gambar untuk *puzzle* (sumber : vecteezy.com)

Untuk membuat fitur *puzzle*, ikuti langkah-langkah berikut :

1. Buka kembali *file* (**tutorial 8 – Tebak Satwa**) dan *file* MPI-component fla.
2. **Klik Frame 110** tambahkan **Blank Keyframe** pada **Layer kode** dan **Layer konten**, dan tambahkan **Frame** pada **Layer mpi** dan **Layer background**.
3. Untuk menambahkan fitur *puzzle* diperlukan 3 elemen, kepingan *puzzle*, kontainer untuk *puzzle* dan gambar yang akan dijadikan *puzzle*. Buka panel **Library file MPI-Component**, kemudian drag **MovieClip kepingPuzzle** dan **puzzleGameMC**.



Gambar 222. menambahkan elemen *puzzle*

Hapus *MovieClip* kepingPuzzle karena sudah memiliki *Linkage* dan kode MPI akan secara otomatis menggunakannya sebagai acuan untuk memotong gambar. Apabila menginginkan bentuk potongan yang berbeda, maka bentuk yang ada di dalam *MovieClip* kepingPuzzle tersebut dapat diedit sesuai kebutuhan. *MovieClip* kepingPuzzle tersusun secara *Frame by Frame* sebanyak 9 *Frame* untuk menampung 9 potongan *puzzle*.

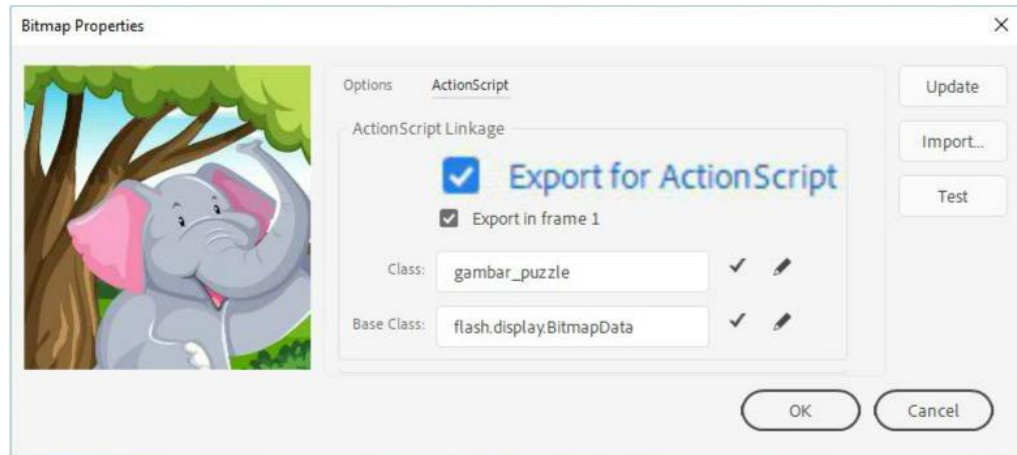
Selanjutnya **klik *MovieClip* puzzleGameMC** dan tambahkan ***instance name* "puzzleMC"**. **Dobel klik *MovieClip*** tersebut, di dalamnya terdapat 2 elemen, *outline* kotak berwarna merah dan tombol **acakBtn**.

Kotak merah tersebut merupakan estimasi ukuran *puzzle* yang akan ditampilkan. Karena gambar yang akan digunakan berukuran 700x400 *pixel*, maka **dobel klik *outline*** merah tersebut, buka panel ***Properties*** dan atur ukurannya menjadi 700x400 *pixel*. Dengan cara ini dapat diketahui estimasi ukuran *puzzle* terhadap *Stage* (layar).

Atur ulang posisi tombol **acakBtn**, pengeditan grafis juga dapat dilakukan pada tombol tersebut.

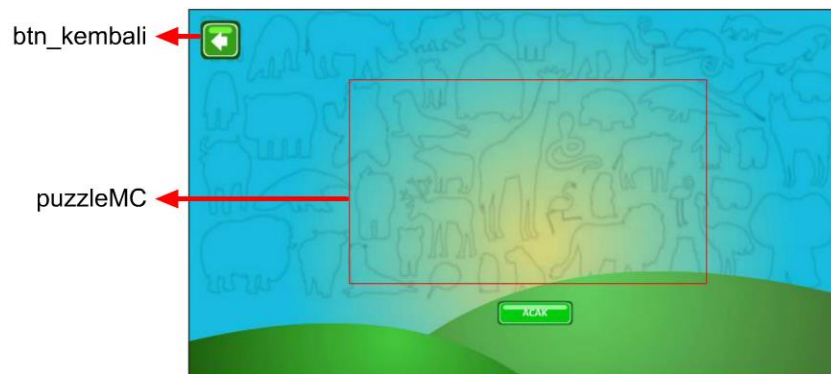
Keluar dari mode edit *symbol*.

4. Tahapan berikutnya adalah menambahkan gambar yang akan dijadikan acuan *puzzle*. **Impor gambar ke *Library***, selanjutnya pada panel *Library* **klik kanan** gambar, dan pilih menu ***Properties***. Pilih panel ***Actionscript*** dan centang opsi ***Export for ActionScript***, pastikan pada kolom *class* tertulis "**gambar_puzzle**".



Gambar 223. menambahkan *Linkage* pada *Bitmap*

5. Tambahkan tombol “**btn_kembali**” pada pojok kiri atas.



Gambar 224. elemen penyusun halaman *puzzle*

6. **Klik *Frame 110 Layer kode***, kemudian tambahkan kode berikut :

```
mpi.click("btn_kembali", 80);
mpi.puzzle("puzzleMC", "gambar_puzzle", selesaiPuzzle);
puzzleMC.lebar = 7; //jumlah kepingan (lebar puzzle)
puzzleMC.tinggi = 4; //jumlah kepingan (tinggi puzzle)
puzzleMC.ukuran = 100; //ukuran kepingan puzzle (pixel)
mpi.updatePuzzle();

function selesaiPuzzle() {
    mpi.popup("popupMC", "Puzzle Selesai", "Selamat!! Anda
    berhasil menyelesaikan puzzle");
    puzzleMC.acakBtn.visible = true;
}
```

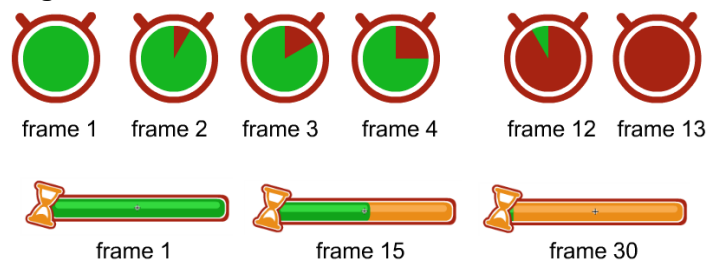
7. Jalankan aplikasi untuk menguji fitur *puzzle*.



Gambar 225. tampilan fitur *puzzle*

8.7 Timer

Timer digunakan sebagai indikator waktu untuk membatasi sebuah kegiatan tertentu. Sebagai contoh memberikan waktu sekian detik untuk menjawab kuis, atau untuk menyelesaikan *puzzle*. *Timer* dalam *file* MPI-component menggunakan visualisasi *MovieClip* yang terdiri dari beberapa *Frame* sebagai indikator waktu.



Gambar 226. *MovieClip* timerMC

Pada contoh di atas, terdapat 2 contoh visualisasi *MovieClip* timer. Contoh di atas menggunakan teknik *Frame by Frame* yang menunjukkan waktu dalam format lingkaran yang semakin lama semakin habis (menggunakan 13 *Frame*). Sementara contoh ke-dua menggunakan teknik *motion tween* (*classic tween*) mulai dari waktu penuh sampai waktu habis. Teknik apapun yang digunakan dan berapapun durasi *MovieClip* dapat dideteksi oleh kode, yang terpenting adalah dimulai dari waktu penuh sampai dengan waktu habis.

Pada contoh berikut akan ditambahkan fitur *timer* pada halaman kuis. Ikuti langkah-langkah berikut:

1. Buka kembali file (**tutorial 8 – Tebak Satwa**) dan file *MPI-component.fla*.
2. Klik **Frame 100 Layer konten**. Buka **Library file MPI-Component** kemudian drag *MovieClip timerMC2* ke *Stage*. Letakkan di pojok kanan atas dan tambahkan **instance name** “*timerMC*”.

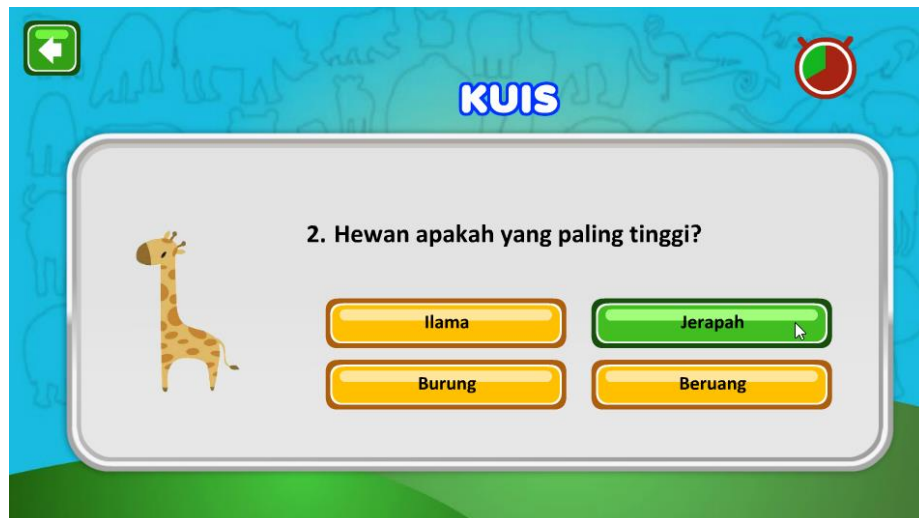


Gambar 227. menambahkan *MovieClip* timerMC

3. Klik **Frame 100 Layer kode**, kemudian tambahkan kode berikut :

```
//menjawab dengan waktu maksimal 5 detik
mpi.timer("timerMC", 5);
```

4. Jalankan aplikasi untuk menguji fitur *timer* pada halaman kuis.



Gambar 228. hasil penambahan fitur *timer*

Penambahan kode `mpi.timer("timerMC", 5)` secara otomatis akan terintegrasi dengan sistem kuis dengan menambahkan waktu menjawab setiap soal selama 5 detik. Kode *timer* juga dapat terintegrasi secara otomatis dengan kode `mpi.puzzle`. Parameter lain yang dapat ditambahkan pada kode *timer* adalah fungsi setelah *timer* habis, contoh :

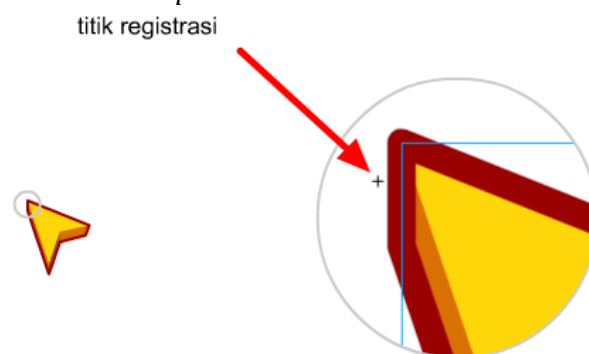
```
mpi.timer("timerMC", 30, fungsiWaktuHabis);
```

pada contoh di atas, setelah waktu mencapai 30 detik, maka `fungsiWaktuHabis` akan dijalankan. Sedangkan kode untuk menghentikan *timer* adalah sebagai berikut :

```
timerMC.aktif = false;
```

8.8 Kursor Mouse

Pada beberapa aplikasi untuk desktop atau *web* seringkali ditemui kursor *mouse* dengan bentuk yang berbeda. Untuk membuat fitur tersebut terlebih dahulu dibuat sebuah *MovieClip* dengan *Linkage* "**kursorMC**". Perlu diperhatikan bahwa dalam membuat kursor, tidak boleh ada grafis yang menutupi titik registrasi, karena dapat menyebabkan deteksi tombol tidak berhasil. Perhatikan contoh *MovieClip* kursor berikut :



Gambar 229. *MovieClip* kursorMC

Untuk mengaktifkan kursor tersebut, **klik *Frame 2 Layer* kode** kemudian tambahkan kode berikut :

```
mpi.kursor("kursorMC");
```

Untuk menonaktifkan kursor, dan mengembalikan kursor *default*, maka dapat dituliskan kode:

```
mpi.kursor("");
```

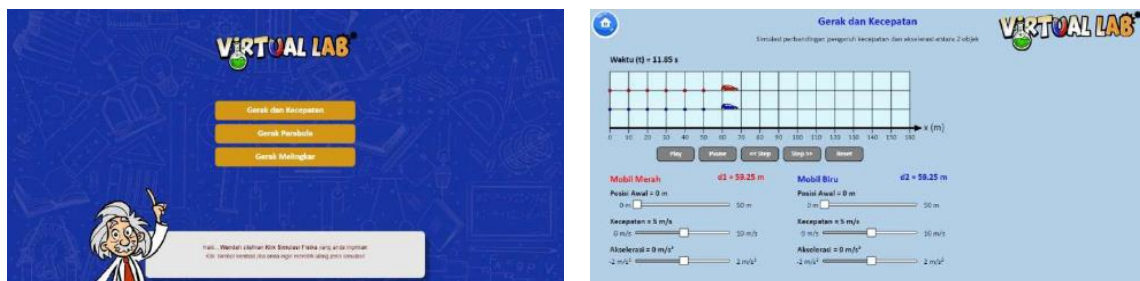
BAB 9

Virtual Lab

9.1 Laboratorium Virtual

Laboratorium virtual dapat didefinisikan sebagai perangkat lunak multisensori yang memiliki interaktivitas untuk mensimulasikan praktikum-praktikum tertentu dengan mereplikasi laboratorium konvensional. Laboratorium virtual memungkinkan pengguna aplikasi untuk belajar melalui pendekatan studi kasus, berinteraksi dengan peralatan laboratorium, melakukan eksperimen, menganalisis eksperimen sekaligus mengevaluasi proses yang dilakukan dalam ruang non fisik.

Laboratorium virtual secara mendetail telah dibahas pada buku sebelumnya, yaitu buku “Laboratorium Virtual: konsep dan pengembangan simulasi fisika”, yang dapat diunduh di situs www.wandah.org/Book. Secara spesifik buku sebelumnya membahas pengembangan simulasi berbasis HTML5 dan Javascript dengan luaran *web based*.



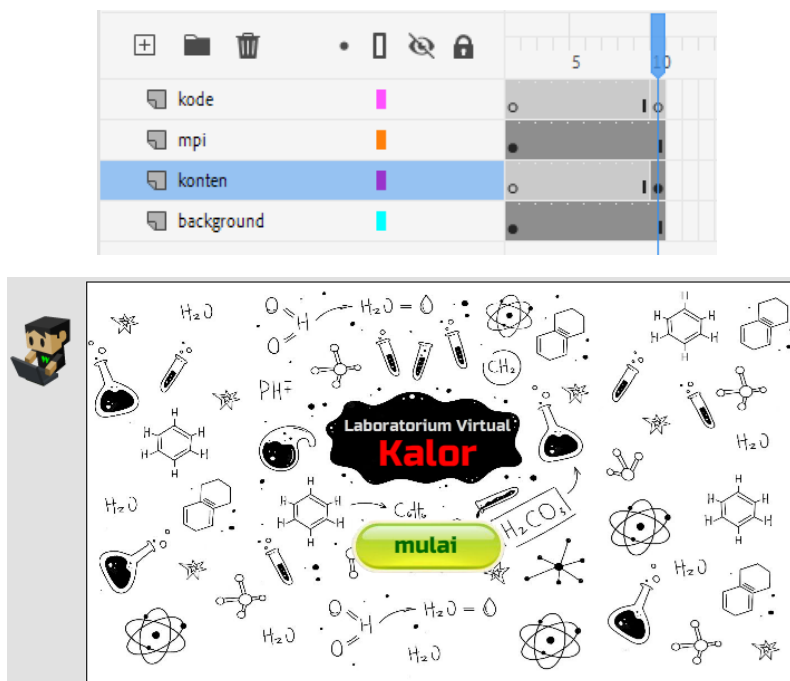
Gambar 230. Laboratorium Virtual berbasis HTML 5 Canvas

Pada bab ini akan dijelaskan tentang pengembangan laboratorium virtual menggunakan *MPI Component*. Pengembangan simulasi perangkat laboratorium menggunakan objek-objek yang telah tersedia di dalam *Library* dan menggunakan kode yang relatif sederhana apabila dikomparasi dengan kode yang digunakan untuk mengembangkan simulasi yang sama dengan menggunakan Javascript / HTML5.

9.1.1 Menyiapkan Halaman Simulasi

Untuk memulai pembuatan laboratorium virtual terlebih dahulu disiapkan halaman yang berfungsi untuk menavigasi pengguna sampai menuju simulai yang dimaksud. Perhatikan langkah-langkah berikut :

1. Buatlah sebuah **file baru (tutorial 9)** dengan pengaturan ukuran 1280x720 *pixel*, 30 FPS dan *Action Script 3.0*.
2. Buatlah **4 buah Layer** dan ubah nama masing-masing *Layer* menjadi *Layer* “background”, “konten”, “mpi” dan “kode”.
3. Klik **Frame 10**, kemudian tambahkan **Blank Keyframe** pada *Layer* **kode** dan *Layer* **konten**. Tambahkan **Frame** pada *Layer* **mpi** dan *Layer* **background**.
4. Buka **file MPI-Component**. Kembali ke **file** proyek yang sedang dikerjakan. Klik **Frame 1 Layer mpi** kemudian *drag* objek **MPI** dari **Library MPI-Component** ke *Stage*.
5. Klik **Frame 10 Layer konten**, kemudian buatlah sebuah tampilan halaman judul berikut sebuah **tombol** dengan *instance name* “**btn_mulai**”. Tambahkan juga latar agar desain menjadi menarik.



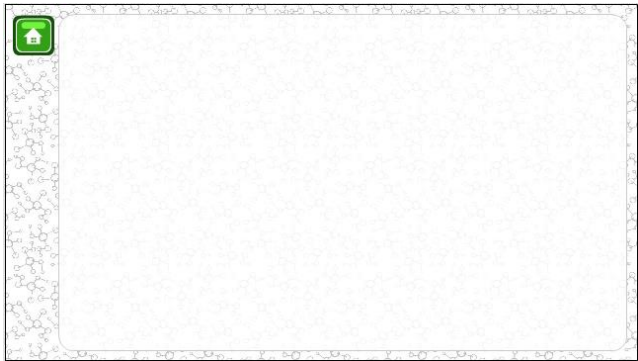
Gambar 231. struktur *Timeline* dan desain halaman judul

6. Klik **Frame 10 Layer kode**, kemudian ketikkan kode berikut :

```
stop();
mpi.click("btn_mulai", 20);
```

7. Pada tahapan berikutnya ditambahkan halaman untuk menampilkan simulasi, yaitu pada **Frame 20**. Klik **Frame 20**, kemudian tambahkan **Blank Keyframe** pada *Layer* **kode**, *Layer* **konten**, dan *Layer* **background**. Tambahkan **Frame** pada *Layer* **mpi**.

Ubah **background** menjadi lebih sederhana (agar tidak mengganggu tampilan simulasi yang nantinya akan ditambahkan. Selanjutnya tambahkan **tombol “btn_home”** untuk kembali ke halaman judul.



Gambar 232. konten *Frame 20*

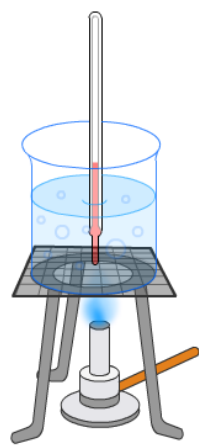
8. Klik **Frame 20 Layer kode**, kemudian ketikkan kode berikut :

```
mpi.click("btn_home", 10) ;
```

9. **Simpan** dengan nama **tutorial-9**. *File* ini selanjutnya akan dijadikan *template* untuk membuat simulasi-simulasi laboratorium virtual.

9.2 Simulasi Panas

Rancangan simulai yang akan dibuat adalah percobaan perubahan suhu akibat perpindahan kalor. Percobaan di laboratorium riil melibatkan alat pembakar spiritus, gelas *beaker*, kaki tiga (*tripod*), termometer dan static untuk meletakkan termometer. Percobaan ditujukan untuk mengetahui perubahan suhu pada beberapa jenis zat cair seperti air dan minyak. Perubahan suhu dicatat seiring dengan variabel waktu dan volume zat.



Nama Zat						
Volume (ml)						
Waktu (Menit)	0,5	1	1,5	2	2,5	3
Suhu (°C)						

Gambar 233. percobaan perubahan suhu dan tabel pengamatan

Pada percobaan tersebut, terdapat pengaturan dan rumus yang digunakan, yaitu :

$$Q = m \cdot c \cdot \Delta T$$

Dimana : Q = Kalor yang diperlukan untuk menaikkan suhu,
 m = massa benda
 c = kalor jenis benda
 ΔT = perubahan suhu

Dengan memahami rumus tersebut, maka percobaan dapat disimulasikan dalam bentuk virtual/digital dengan menggunakan model matematis dan pemanfaatan simbol-simbol yang ada.

Perlu dipahami terdapat 2 konsep terkait dengan pengembangan simulasi laboratorium virtual dengan menggunakan MPI-component, yaitu

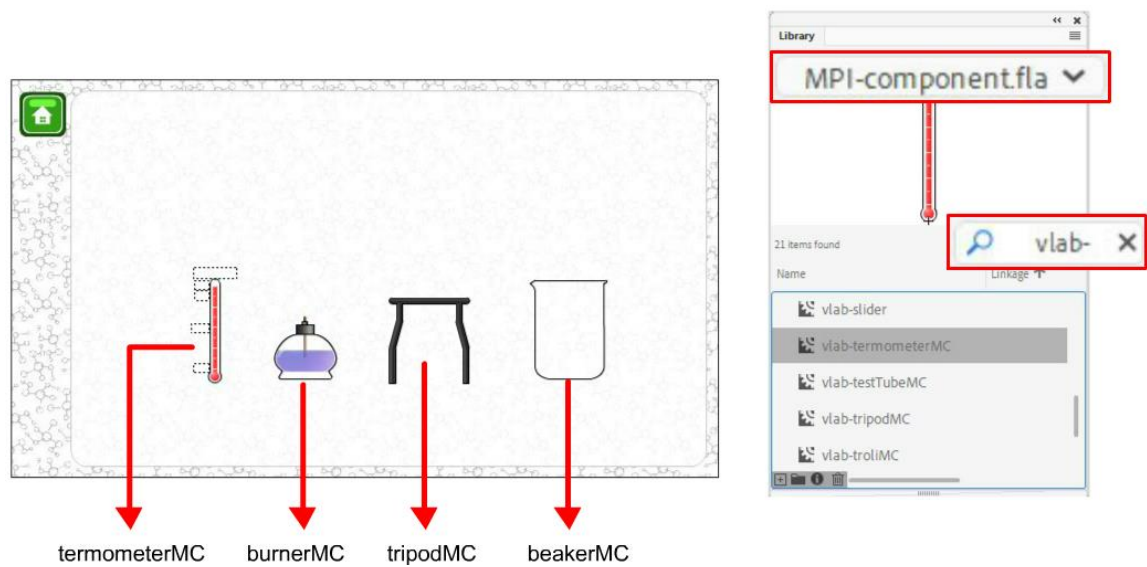
1. Mode otomatis oleh sistem atau oleh kode yang sudah terdapat di dalam *MPI-component*. Dengan menggunakan mode otomatis, tidak diperlukan pengaturan yang kompleks karena kode telah disimulasikan serealistik mungkin, namun dimungkinkan perbedaan perhitungan apabila dikomparasi dengan kondisi riil.

Sebagai contoh dalam memanaskan air dalam gelas kaca, secara otomatis panas dari *burner* akan menaikkan suhu air sesuai dengan volume air, namun dari sisi waktu disimulasikan menjadi lebih cepat, sehingga proses pemanasan atau pendinginan akan berlangsung lebih cepat dari kondisi riil.

2. Mode manual, yaitu pengaturan secara otomatis akan dimatikan, dan *component* yang ada akan berfungsi sebatas media visualisasi dari simulasi yang akan dilakukan. Pengembang membutuhkan kode dan permodelan matematis untuk mendapatkan simulasi yang diinginkan. Dengan metode ini simulasi yang dihasilkan dapat mengacu pada kondisi riil, namun membutuhkan pengetahuan khusus dalam pemrograman.

Perhatikan contoh berikut, untuk membuat simulasi perubahan suhu menggunakan mode otomatis *mpi-component* :

1. Buka kembali *file* (**tutorial 9**), kemudian pilih menu **save as** untuk menyimpannya dengan nama lain. Simpan dengan nama **tutorial 9 – simulasi panas**.
2. Buka *file* **MPI-component**.
3. Kembali ke *file* **tutorial 9 – simulasi panas**. Klik **Frame 20 Layer konten**, selanjutnya *drag* beberapa *symbol* yang dibutuhkan untuk membuat simulasi. Tambahkan **instance name** pada masing-masing *symbol* tersebut (sesuai dengan gambar). Sesuaikan ukuran *symbol* dengan menggunakan *transform tool*.



Gambar 234. *symbol* untuk membentuk simulasi

4. Klik **Frame 20 Layer kode**, kemudian tambahkan kode berikut :

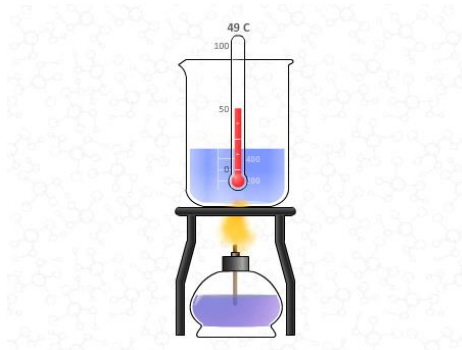
```
mpi.glass("beakerMC");
beakerMC.volume = 50;

mpi.tripod("tripodMC");
tripodMC.dragY = false;

mpi.burner("burnerMC");
burnerMC.dragY = false;
burnerMC.menyala = true;

mpi.termometer("termometerMC");
```

5. Jalankan aplikasi untuk mengujicoba simulasi. Pada saat ini, seluruh objek dapat berinteraksi, dan ketika *glass beaker* diletakkan di atas api dan dilakukan pengukuran suhu dengan termometer.



Gambar 235. Hasil simulasi

9.3 Interaktivitas pada Simulasi

Simulasi yang sudah dibuat pada tutorial sebelumnya memiliki interaktivitas yang rendah, karena tidak ada pengaturan dinamis yang dapat dilakukan. Pengaturan dinamis yang dimaksud adalah kemampuan pengguna untuk mengatur berbagai variabel sehingga pengguna dapat mengambil kesimpulan lebih baik terhadap simulasi yang sedang dijalankan. Pada contoh berikut akan diberikan pengaturan pada beberapa elemen simulasi.

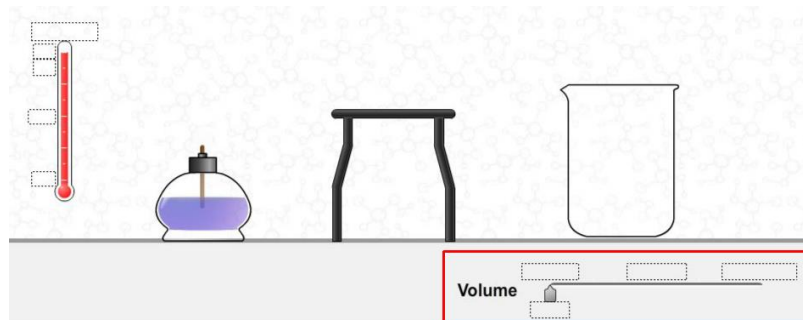
9.3.1 Mengatur Volume Zat Cair dengan *Slider*

Salah satu variabel yang berpengaruh dalam perubahan suhu adalah massa benda, atau dalam hal ini dipengaruhi oleh volume benda. Pada contoh di atas, volume *beaker* diatur dengan menggunakan kode :

```
beakerMC.volume = 50;
```

Untuk mendapatkan pengaturan volume yang lebih dinamis perlu ditambahkan fitur *slider*. Perhatikan langkah-langkah berikut :

1. Lanjutkan *file tutorial 9 – simulasi panas*.
2. Klik **Frame 20 Layer konten**, buka *Library MPI-Component* dan drag *MovieClip vlab-slider*. Tambahkan *instance name “sliderMC”*.



Gambar 236. penambahan *slider*

3. Klik **Frame 20 Layer kode** dan tambahkan kode berikut untuk mengaktifkan *slider*.

```
mpi.slider("sliderMC", aturVolume);  
sliderMC.nilai = 50;  
  
function aturVolume() {  
    beakerMC.volume = sliderMC.nilai;  
    mpi.updateMC("beakerMC");  
}
```

4. Jalankan aplikasi untuk mengujicoba *slider*.

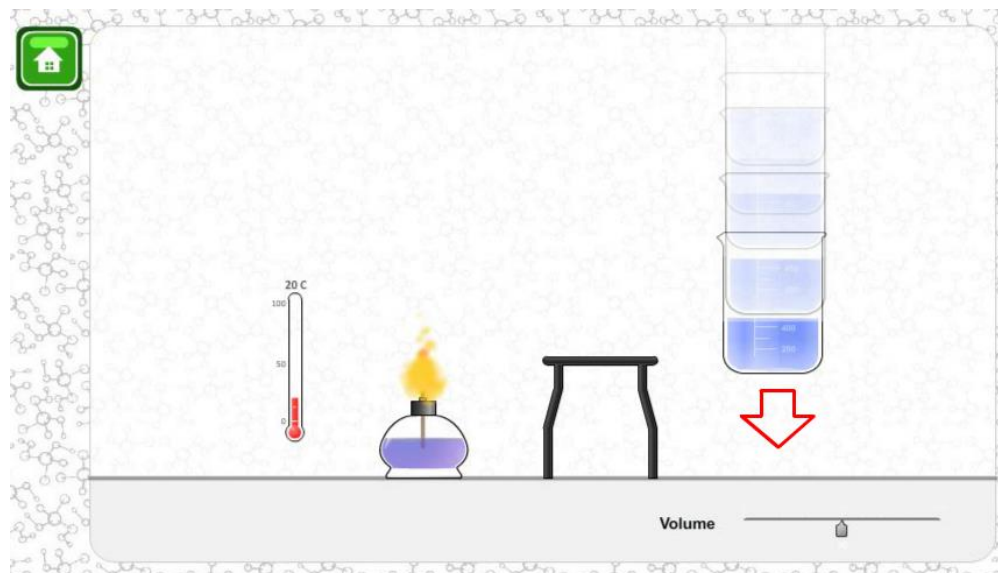
Pada contoh di atas, kode `mpi.slider` secara otomatis akan menampilkan nilai *default* yaitu antara 0 – 100. Kode `mpi.slider` dapat diatur memiliki nilai yang spesifik dengan menambahkan kode pengaturan sebagai berikut :

```
sliderMC.nilai = 50; //mengatur nilai awal slider
sliderMC.kelipatan = 1; //mengatur kelipatan pergeseran
sliderMC.desimal = 2; //mengatur jumlah angka di belakang koma
sliderMC.nilaiMin = 0; //nilai minimal slider
sliderMC.nilaiMax = 100; //nilai maksimal slider
```

9.3.2 Menambahkan Gravitasi pada Objek

Untuk menambah kesan realistis sebuah objek dalam simulasi, dapat ditambahkan efek gravitasi. Kode yang digunakan untuk menambahkan gravitasi adalah sebagai berikut :

```
beakerMC.gravitasi = true; //gravitasi aktif ketika bernilai true
mpi.vlabLantai = 595; //koordinat Y objek maksimal
```



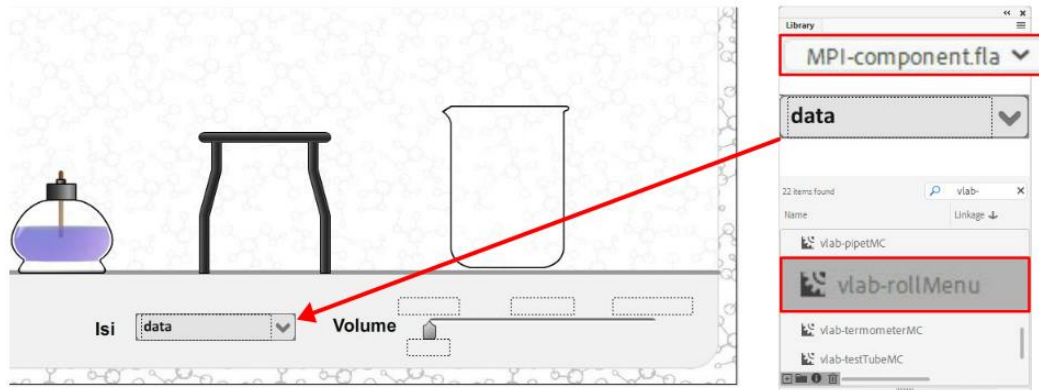
Gambar 237. hasil penambahan efek gravitasi

9.3.3 Menambahkan Pengaturan Isi Gelas

Pada beberapa percobaan untuk mengetahui tingkat perubahan suhu, diperlukan beberapa zat yang berbeda untuk diujicoba. Sebagai contoh dalam percobaan di atas, pada laboratorium riil sering kali digunakan zat cair yang berbeda seperti air, air garam, alkohol, minyak dan sebagainya. Untuk menambahkan pengaturan tersebut di dalam simulasi dapat ditambahkan sistem menu.

Perhatikan langkah-langkah berikut :

1. Lanjutkan *file tutorial 9 – simulasi panas*.
2. Klik **Frame 20 Layer konten**, buka **Library MPI-Component** dan *drag MovieClip vlab-rollMenu*. Tambahkan **instance name “isiMC”**.



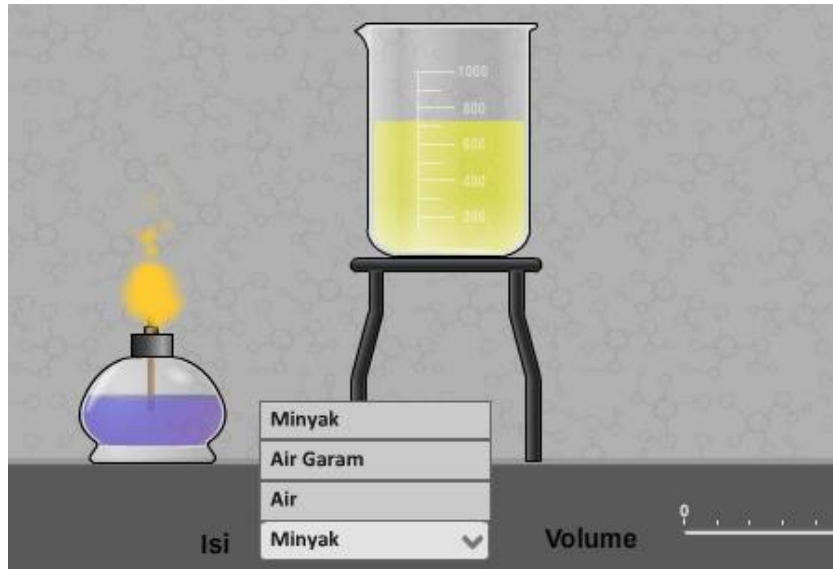
Gambar 238. menambahkan *rollmenu*

3. Klik **Frame 20 Layer kode**, kemudian tambahkan kode berikut :

```
//roll menu untuk mengubah isi gelas
isiMC.data1 = "Air";
isiMC.data2 = "Air Garam";
isiMC.data3 = "Minyak";
mpi.rollMenu("isiMC", ubahIsi);

function ubahIsi() {
    if (isiMC.pilihan == 1) {
        beakerMC.warna = "biru";
        beakerMC.kalorJenis = 1;
    }
    if (isiMC.pilihan == 2) {
        beakerMC.warna = "putih";
        beakerMC.kalorJenis = 0.93;
    }
    if (isiMC.pilihan == 3) {
        beakerMC.warna = "kuning";
        beakerMC.kalorJenis = 0.6;
    }
    mpi.updateMC("beakerMC");
}
```

4. Jalankan aplikasi untuk mengujicoba fitur perubahan isi gelas dan efek perubahan suhu akibat perbedaan kalor jenis.



Gambar 239. hasil penambahan *rollMenu*

9.4 Penambahan Rumus pada Simulasi

Pada contoh sebelumnya, simulasi berjalan secara otomatis. Beberapa variabel seperti kalor jenis, volume air, dan kalor yang dikeluarkan oleh *burner* disimulasikan sedemikian rupa. Namun dalam percobaan yang sesungguhnya dimungkinkan adanya perbedaan data hasil penelitian, karena dalam simulasi tersebut perhitungan yang dilakukan di dalam kode *MPI Component* bisa jadi meleset karena adanya perubahan satuan waktu yang digunakan. Dalam simulasi otomatis yang ada dalam *MPI Component*, waktu dimanipulasi menjadi lebih cepat.

Apabila pengembang ingin mendapatkan hasil pengukuran yang tepat, sesuai dengan kondisi riil percobaan di laboratorium yang sesungguhnya, maka perlu dilakukan perhitungan rumus secara manual. Sebagai contoh, untuk memperoleh perubahan suhu pada zat cair yang berada di gelas, dapat digunakan rumus :

$$Q = m \cdot c \cdot \Delta T$$

$$\Delta T = Q / (m \cdot c)$$

Perhatikan langkah-langkah berikut, untuk menerapkan rumus tersebut ke dalam aplikasi laboratorium virtual:

1. Buka *file tutorial 9 – simulasi panas*, kemudian **save as** dengan nama **tutorial 9 – rumus simulasi panas**.
2. Klik **Frame 20 Layer konten** tambahkan tombol **“btn_mulaiSimulasi”** dan tombol **“btn_resetSimulasi”** . Tombol ini akan digunakan untuk mengatur simulasi.
3. Klik **Frame 20 Layer kode**, kemudian tambahkan kode berikut :


```

//menggunakan rumus
beakerMC.auto = false;
burnerMC.menyala = false;

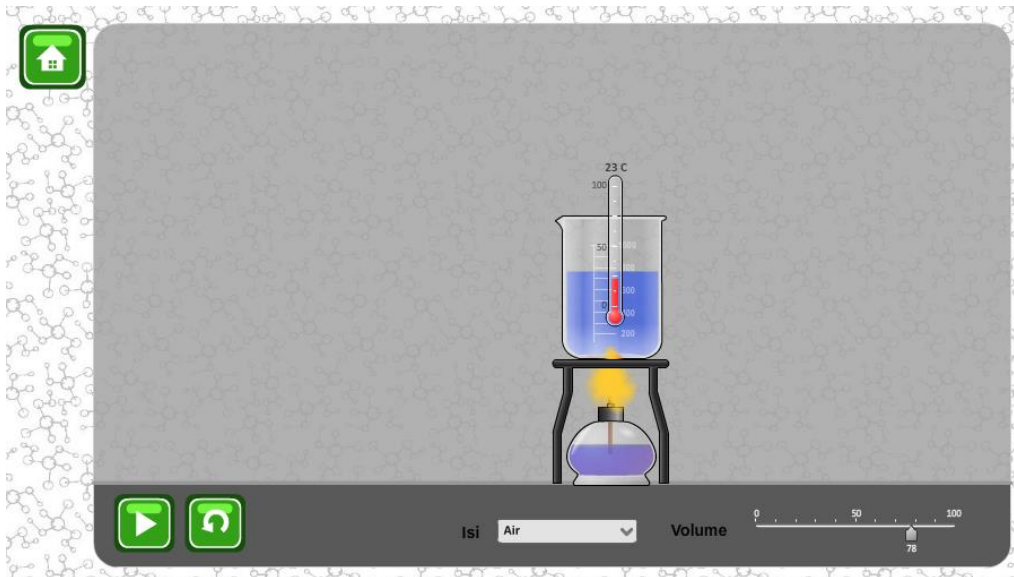
//kalori burner
burnerMC.kalori = 10;

mpi.click("btn_mulaiSimulasi", "start-vlab", rumusSuhu);
mpi.click("btn_resetSimulasi", "reset-vlab");

function rumusSuhu() {
    burnerMC.menyala = true;
    //T = Q / (m.c)
    var kenaikanSuhu =
        burnerMC.kalori / (beakerMC.volume * beakerMC.kalorJenis);
    //jika api menyentuh gelas
    if (beakerMC.sumberPanas == burnerMC) {
        //suhu berubah
        beakerMC.suhu += kenaikanSuhu;
    }
}

```

4. Jalankan aplikasi untuk mengujicoba penerapan rumus.



Gambar 240. hasil simulasi dengan rumus

9.4.1 Penjelasan Program

Untuk memastikan bahwa simulasi menggunakan penambahan rumus secara manual, maka kode yang ada di dalam *MovieClip* beakerMC dinonaktifkan terlebih dahulu menggunakan kode: `beakerMC.auto = false;` Dengan cara ini, perhitungan suhu pada beakerMC tidak dilakukan secara otomatis, sehingga ketika tersentuh api tidak akan mengalami kenaikan suhu.

Selanjutnya `burnerMC` dimatikan terlebih dahulu, dan diatur tingkat kalori yang dilepaskan melalui kode `burnerMC.kalori = 10;`

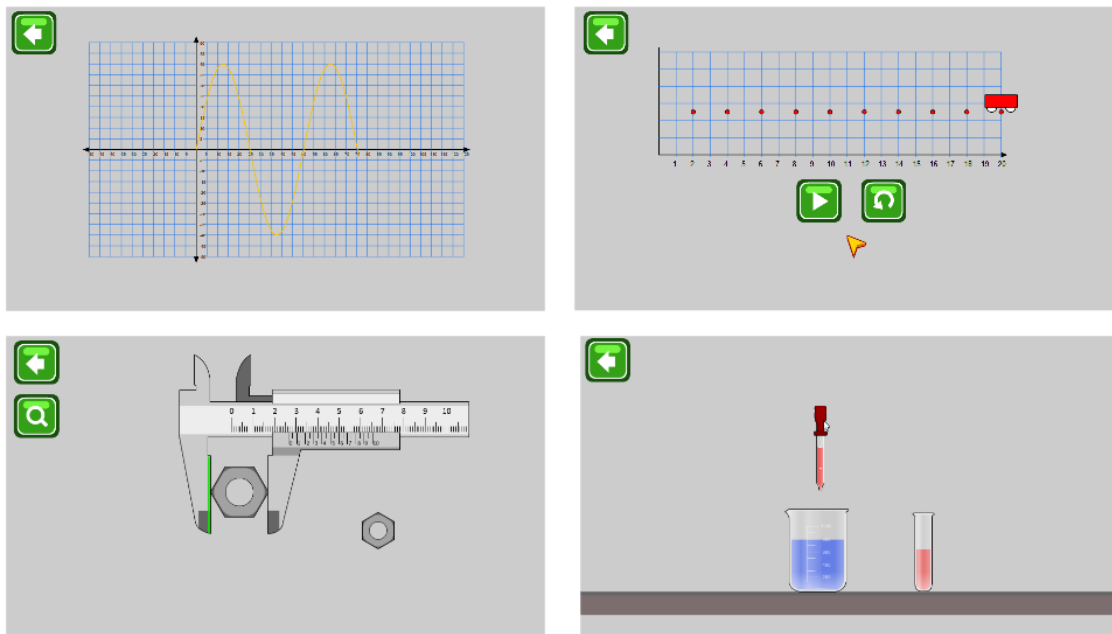
Pada simulasi tersebut juga terdapat 2 buah tombol, yaitu tombol untuk memulai simulasi dengan menjalankan fungsi `rumusSuhu`, serta tombol untuk *merestart* kembali seluruh elemen simulasi seperti kondisi awal.

Pada fungsi `rumusSuhu`, dilakukan pengaturan `kenaikanSuhu` sesuai dengan rumus yang berlaku. Dalam kasus ini volume dianggap sebagai massa, apabila menginginkan perhitungan yang lebih detail, maka dapat ditambahkan variabel masa jenis untuk menghitung massa isi `beakerMC` sesuai dengan `volume` yang ada.

9.5 Pengembangan Simulasi Lain

Pada buku ini disajikan sebuah contoh pengembangan simulasi atau aplikasi untuk laboratorium virtual. Contoh ini digunakan untuk menunjukkan gambaran umum proses pengembangan simulasi dengan menggunakan *MPI-Component*. Untuk pengembangan simulasi lainnya, *MPI Component* telah menyediakan beberapa contoh yang dapat dilihat secara langsung pada *file* yang ada.

File MPI-component dengan dukungan simulasi laboratorium virtual akan selalu *diupdate* untuk menambahkan fitur-fitur baru terkait dengan pengembangan laboratorium virtual. Selain itu, *MPI-component* juga memberikan fleksibilitas kepada pengembang apabila pengembang ingin menambahkan kode *Actionscript* bersamaan dengan kode-kode *MPI-component*.



Gambar 241. beberapa simulasi dalam *file MPI Component*

BAB 10

Publishing Phase

10.1 Konsep Dasar *Multiplatform*

Multiplatform merupakan sebuah istilah yang menunjukkan bahwa sebuah aplikasi dapat dijalankan di beberapa *platform* yang berbeda. Tujuan dari pengembangan aplikasi *multiplatform* adalah untuk meningkatkan jangkauan terhadap target pengguna. Pengembangan *multiplatform* menjadi sebuah standar industri aplikasi, hal ini dikarenakan diversifikasi *platform* yang sangat luas dan masing-masing *platform* populer memiliki pasar yang besar. Katakanlah pasar *mobile apps*, dan secara spesifik *mobile apps* dengan sistem operasi Android memiliki jumlah yang sangat signifikan yaitu 2,3 milyar perangkat di tahun 2022. Ketika sebuah aplikasi mampu menjangkau satu per mil dari pasar tersebut, berarti 2,3 juta aplikasi telah terinstall. Demikian halnya dengan *platform mobile apps* dengan sistem operasi IOS (Apple) yang memiliki jumlah lebih dari 1,2 milyar perangkat per 2022. Pasar Apple dikenal memiliki pengguna yang loyal dan memiliki kemampuan berbelanja yang lebih tinggi dibandingkan pengguna Android, hal ini tentunya juga menjadi pertimbangan dalam mengembangkan aplikasi di *platform* tersebut.

Jumlah pasar *multiplatform* yang sangat besar, didukung dengan terpublikasinya jutaan aplikasi baik aplikasi berbayar maupun aplikasi gratis merupakan sebuah peluang besar dalam menjadikan aplikasi multimedia interaktif mencapai tujuan berikutnya. Setelah fokus terhadap tujuan edukatif dengan materi informatif yang ingin disampaikan kepada pengguna, tujuan kedua adalah tujuan finansial (komersial). Sebuah aplikasi multimedia interaktif dapat dikomersialkan melalui berbagai cara seperti penjualan aplikasi secara langsung, penambahan iklan, pembelian di dalam aplikasi, fitur *freemium*, *sponsorship* dan berbagai metode lainnya. Hal ini tentunya menjadi pembahasan tersendiri karena membutuhkan pengetahuan yang lebih detail tentang komersialisasi aplikasi.

Multiplatform menawarkan peluang yang sangat besar, namun mengembangkan aplikasi yang dapat berjalan dengan baik di berbagai *platform* adalah sebuah tantangan besar. Masing-masing *platform* memiliki keunikan tersendiri, kebutuhan sistem yang beragam, bahasa pemrograman khusus, serta kebijakan publikasi aplikasi yang berbeda. Pengembangan aplikasi dengan teknik *multiplatform framework* menjadi standar industri dalam 2 dekade terakhir, sehingga beberapa aplikasi sudah mengacu pada konsep *multiplatform* tersebut. Termasuk di dalamnya pengembangan aplikasi dengan menggunakan Adobe Animate.

Adobe Animate memiliki kemampuan untuk menghasilkan *file* akhir untuk beberapa *platform* populer seperti untuk PC (desktop), Android, Apple, dan untuk *web browser*. Khusus untuk *web browser*, pada tahun 2021 Adobe secara resmi menghentikan *plugin* Flash Player yang berfungsi untuk menjalankan *file* bertipe SWF pada *web browser*. Meskipun demikian, saat buku ini ditulis terdapat alternatif yang dapat digunakan untuk menjalankan *file* SWF pada *web browser* menggunakan Ruffle yang akan dijelaskan pada sub bab berikutnya. Dengan demikian aplikasi yang dikembangkan dengan Adobe Animate (sebagaimana contoh tutorial yang ada di dalam buku ini), dapat dipublikasikan secara *multiplatform*.

10.2 Desktop (.exe)

Publikasi pertama yang akan dibahas adalah publikasi pada desktop (komputer) dengan sistem operasi Windows. Publikasi akan menghasilkan *file* aplikasi berekstensi EXE yang dapat dijalankan di berbagai komputer dengan sistem operasi windows. Perhatikan langkah-langkah berikut :

1. Buka kembali *file* **tutorial 8 – Tebak Satwa**.
2. Sebelum melakukan proses *publishing* terdapat beberapa pengaturan yang harus dilakukan, diantaranya adalah menjadikan aplikasi pada mode layar penuh (*fullscreen*). Untuk menambahkan fitur tersebut, klik **Frame 1 Layer kode**, kemudian tambahkan kode berikut :

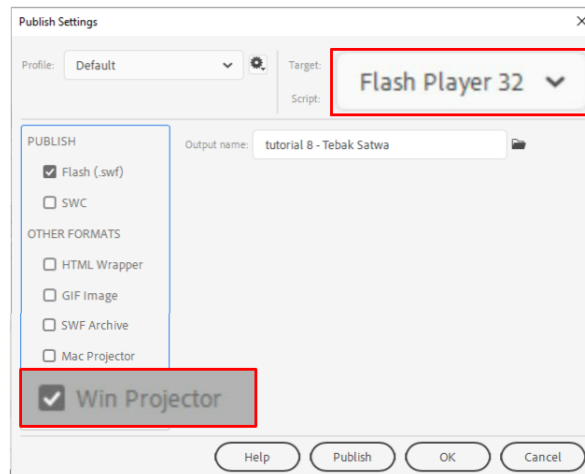
```
Stage.displayState = StageDisplayState.FULL_SCREEN_INTERACTIVE;
```

3. Tahapan berikutnya adalah menambahkan sebuah tombol untuk keluar dari aplikasi. Klik **Frame 10 Layer konten** selanjutnya tambahkan **tombol "btn_keluar"**. Selanjutnya klik **Frame 10 Layer kode** kemudian tambahkan kode :

```
mpi.click("btn_keluar", keluar);

function keluar() {
    fscommand("quit");
}
```

4. Klik menu **File > Publish Setting**. Pada opsi **Target** pilih **Flash Player 32**, ini adalah versi terakhir yang dirilis oleh Adobe. Selanjutnya pada opsi **Other Formats** centang opsi **Win Projector**. Tekan tombol **Publish**.



Gambar 242. pengaturan publikasi desktop

- Setelah beberapa saat, proses publikasi akan selesai dan menghasilkan *file* dengan ekstensi EXE yang siap untuk dijalankan di berbagai komputer. Agar fitur intro dan video dapat dijalankan di manapun, maka *file* **intro.swf**, **video skin.swf** dan *file* **video** perlu diikut sertakan setiap *file* exe dipindahkan.



Gambar 243. proses *publishing* dan *file* yang dihasilkan

10.3 Android

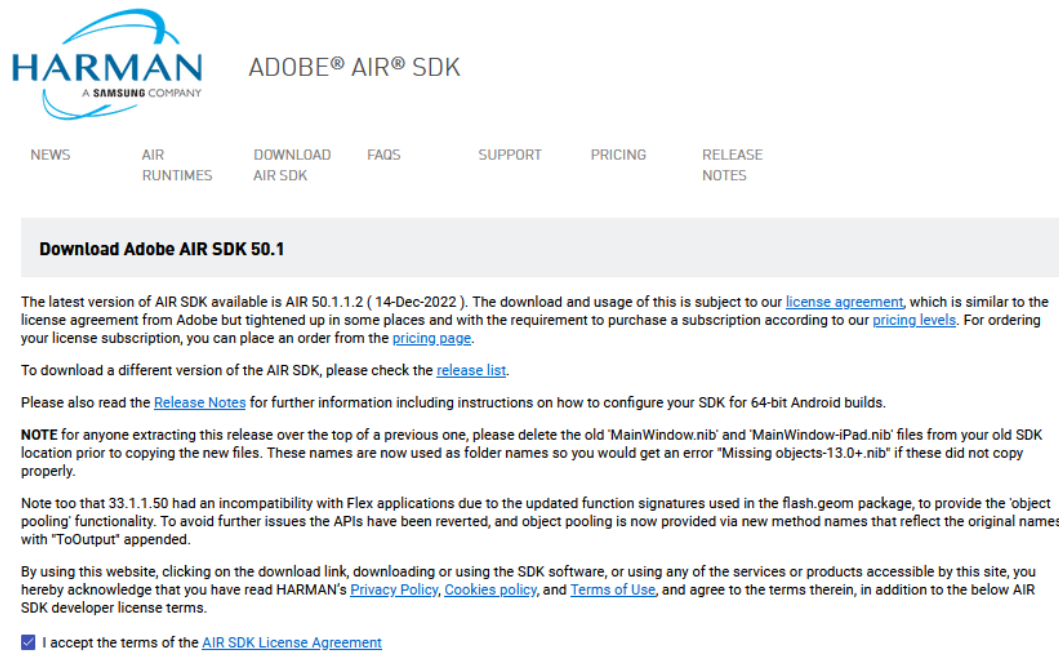
Android merupakan *platform* yang sangat populer, sehingga mempublikasikan aplikasi multimedia interaktif ke perangkat Android akan memiliki jangkauan pengguna yang sangat luas. *File* yang dihasilkan oleh proses *publishing* dengan *platform* Android berekstensi APK. APK merupakan singkatan dari *Application Package File*, standar yang digunakan untuk menginstall aplikasi ke perangkat Android. Meskipun demikian, kebijakan baru Google Playstore adalah menggunakan *file* bertipe AAB (*Android App Bundle*). Pada sub bab berikut akan dijelaskan proses *publishing* untuk *platform* Android baik bertipe APK maupun AAB.

10.3.1 AIR SDK

Dalam mengembangkan aplikasi untuk *platform* Android diperlukan sebuah modul tambahan yaitu Adobe AIR. Adobe® AIR® adalah sistem multi-operasi yang memungkinkan pengembang untuk membangun aplikasi *Rich Internet Application* (RIA) ke *platform* desktop dan perangkat seluler (*mobile*). Aplikasi yang dihasilkan akan bekerja dengan sistem yang sama dengan aplikasi *native* dan dapat dikemas dengan *captive AIR runtime*. *Runtime* menyediakan kerangka kerja dan *platform* sistem lintas operasi (*multiplatform*) yang konsisten sehingga akan mempermudah pengembang mencapai standar industri terbaru.

Sistem operasi Android melakukan *update* secara periodik, yang ditujukan untuk memperbaiki kekurangan (*bug*), penyesuaian terhadap perangkat keras baru atau menambah fitur baru. Oleh karena itu sebuah *framework* yang ditujukan pada *platform* Android juga harus diupdate sedemikian rupa. Demikian halnya dengan AIR SDK yang juga membutuhkan *update* secara periodik. Saat buku ini ditulis, versi AIR SDK terbaru adalah versi 50.1 versi ini akan selalu berubah mengikuti perubahan versi sistem Android, sehingga pengembang harus memastikan menggunakan update versi terbaru untuk mendapatkan performa yang paling optimal. Untuk menambahkan AIR SDK ke dalam Adobe Animate, ikuti langkah-langkah berikut :

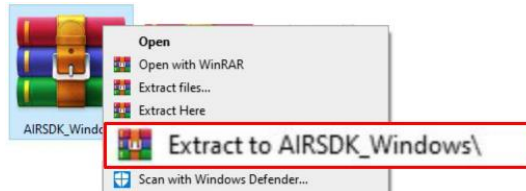
1. Buka situs <https://airsdk.harman.com/download>. AIR SDK dikelola oleh Harman, sebuah perusahaan aplikasi di bawah perusahaan Samsung. Pada halaman tersebut terdapat perjanjian penggunaan lisensi. AIR SDK gratis untuk penggunaan secara personal dan penggunaan secara komersial dengan pendapatan di bawah 100 ribu dolar per tahun. **Centang** persetujuan untuk melanjutkan proses download.



Gambar 244. halaman persetujuan Adobe Air SDK

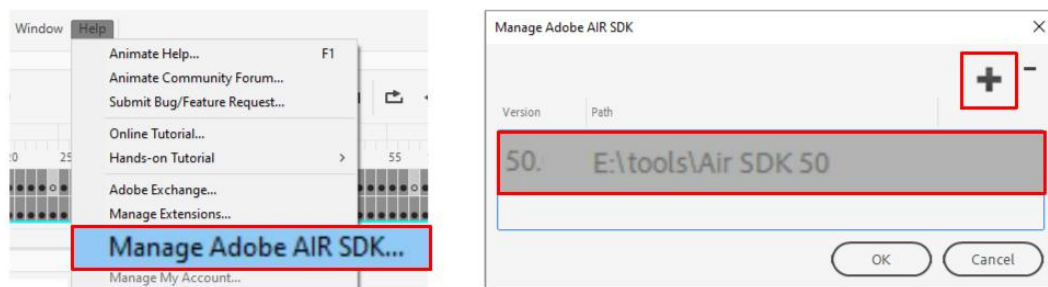
2. Pada bagian bawah terdapat *link* untuk *download* SDK. Klik **AIR SDK for Windows**, maka akan terjadi proses *download* dengan estimasi ukuran download 650 MB.

File yang terunduh dalam kondisi terkompresi dengan ekstensi *file* ZIP. **Klik kanan** *file* tersebut dan **Extract** ke *folder* khusus. Untuk mempermudah proses pengembangan, letakkan beberapa perangkat lunak untuk pengembangan aplikasi di *folder* khusus, misal : E:\Tools\AIRSDK_Windows\



Gambar 245. mengekstrak *file* AIR SDK

3. Buka aplikasi Adobe Animate. Klik menu **Help > Manage Adobe AIR SDK**. Klik tombol + untuk menambahkan **Path** lokasi mengekstrak AIR SDK. Pilih *folder* tempat ekstraksi, apabila berhasil akan muncul versi SDK yang terinstall.



Gambar 246. menambahkan AIR SDK ke dalam Adobe Animate

Sampai dengan tahapan ini, Adobe Animate akan memiliki kemampuan untuk *publish* proyek ke *platform* Android.

10.3.2 Publikasi dengan Output APK

Setelah AIR SDK ditambahkan ke dalam aplikasi Adobe Animate, maka aplikasi memiliki kemampuan untuk mentargetkan *platform* Android (*File* Bertipe APK). Untuk lebih jelasnya perhatikan langkah-langkah berikut :

1. Buka kembali *file* **tutorial 8 – Tebak Satwa**.
2. Sebelum melakukan proses *publishing* Android terdapat beberapa pengaturan yang harus diubah, khususnya untuk tombol keluar. Klik **Frame 10 Layer kode** kemudian ubah kode untuk tombol keluar menjadi sebagai berikut :

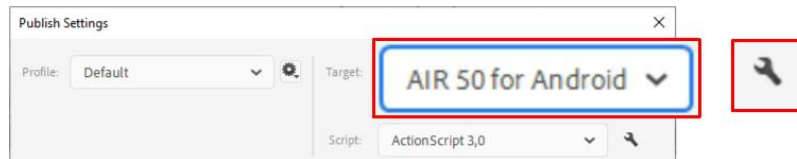
```

mpi.click("btn_keluar", keluar);

function keluar() {
    NativeApplication.nativeApplication.exit();
}

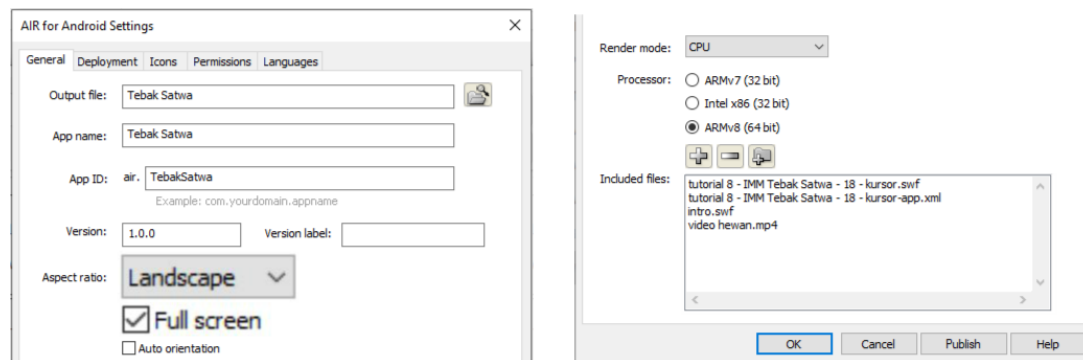
```

3. Klik menu **File > Publish Setting**. Pada opsi **Target** pilih **AIR for Android**. Sebelum menekan tombol *publish*, terdapat beberapa pengaturan yang harus dilakukan terlebih dahulu. Tekan tombol **Setting** untuk membuka pengaturan yang lebih detail.



Gambar 247. memilih target AIR for Android

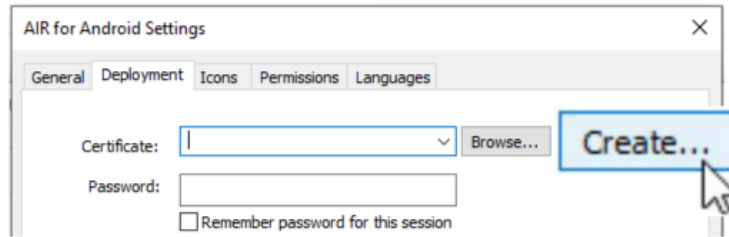
Pada panel **General**, pastikan nama aplikasi sudah sesuai. Pilih **Aspect ratio : Landscape** dan centang opsi **Full screen** agar aplikasi nantinya ditampilkan secara penuh di layar. Pada opsi **Render mode**, pilih opsi **CPU**, opsi ini akan membebankan operasi pada *processor* utama perangkat Android. Apabila terdapat grafik yang intens, maka dapat digunakan opsi **GPU**. Sedangkan opsi **Direct** digunakan untuk menampilkan fitur grafis 3 Dimensi.



Gambar 248. pengaturan panel *General*

Pada kolom **included files**: perlu ditambahkan *file intro.swf*, *file video* dan *file skin video.swf*. Tekan tombol + untuk menambahkan *file-file* tersebut.

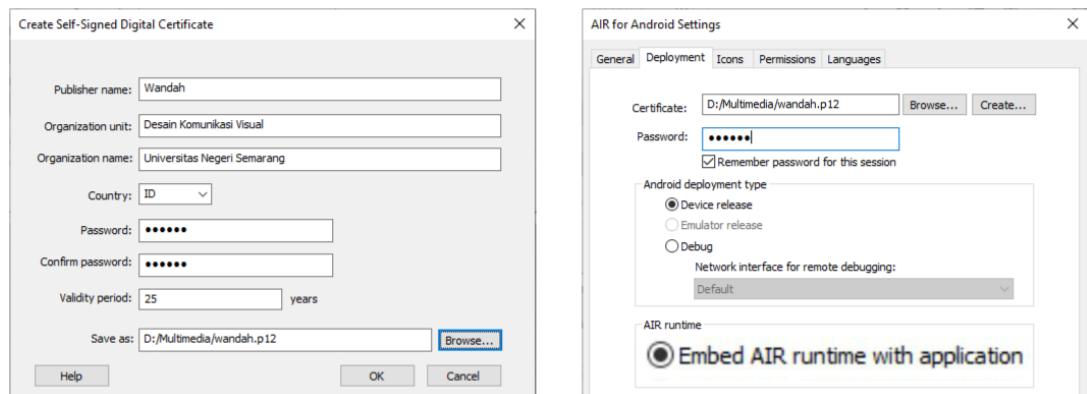
4. Bergeser ke panel **Development**, di dalamnya terdapat opsi **certificate**. Certificate merupakan sebuah *file* bertipe p12 yang merupakan identitas dari pengembang. Apabila belum pernah membuat *certificate*, maka tahapan yang harus dilakukan adalah menekan tombol **create**.



Gambar 249. membuat dokumen *certificate*

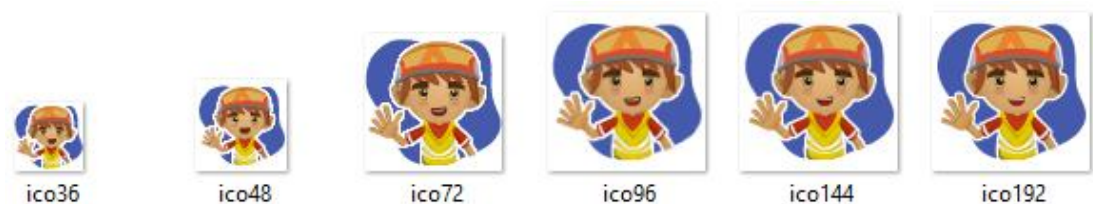
Pada panel pembuatan *certificate*, isilah seluruh *form*, buatlah *password* dan kemudian tekan tombol **Browse**, ketikkan nama *file certificate* dan tekan **OK**. Beberapa detik kemudian *certificate* akan terbentuk dan panel kembali ke menu *Deployment*. Ketikkan *password* dan centang opsi **Remember password**, agar tidak perlu mengetikkan *password* setiap kali menekan tombol *publish*.

Opsi terpenting yang harus tetap dibiarkan dalam kondisi aktif adalah **Embed AIR runtime with application**.



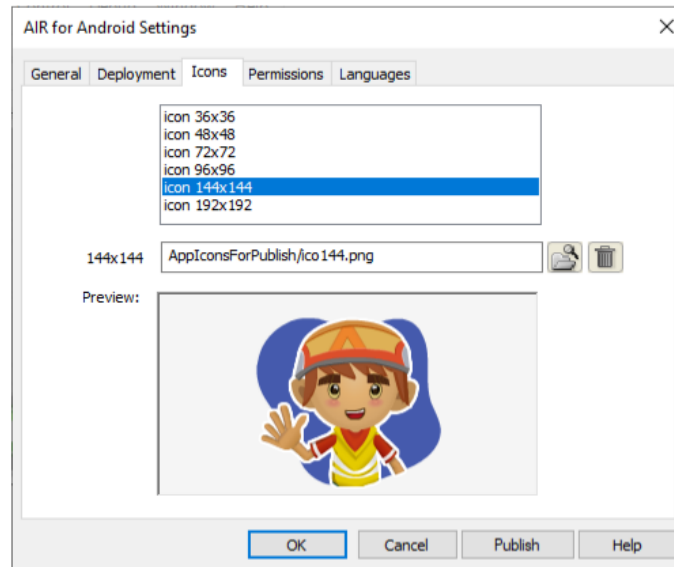
Gambar 250. pengaturan *certificate*

5. Bergeser ke panel berikutnya yaitu **Icons**. *Icons* adalah sebuah simbol atau logo yang akan ditampilkan pada perangkat Android, setelah aplikasi terinstall. *Icons* dapat diisi dengan *file* bertipe PNG yang telah disiapkan sebelumnya. Pastikan ukuran ikon sesuai dengan kolom yang ada.



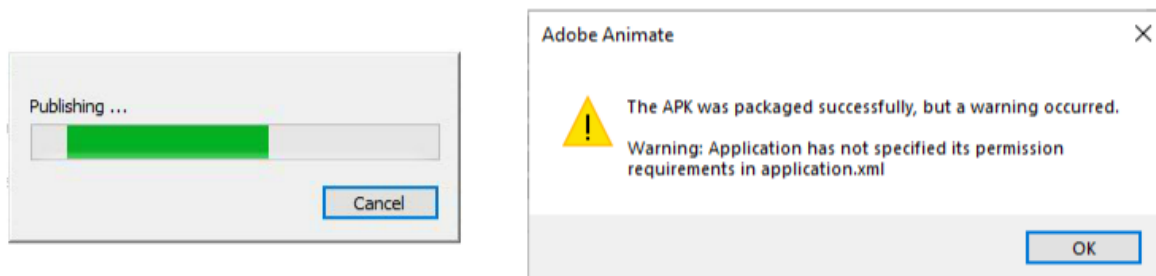
Gambar 251. persiapan *file* icon

Set masing-masing icon sesuai dengan ukuran yang ditentukan. *File* yang dijadikan icon, secara otomatis akan dipindahkan ke dalam *folder AppIconsForPublish*.



Gambar 252. pengaturan Icon

6. Pada panel selanjutnya, yaitu panel **Permissions** dan panel **Languages**, tidak terlalu banyak yang harus diatur karena aplikasi tidak membutuhkan perizinan khusus. Sementara untuk kolom bahasa, dapat dipilih bahasa Inggris.
7. Tekan tombol **Publish**, maka proses publikasi akan berlangsung beberapa detik dan ketika selesai akan menghasilkan *file* bertipe APK. Apabila ada peringatan tertentu seperti aplikasi tidak memiliki perizinan khusus, dapat diabaikan karena hal tersebut berarti aplikasi tidak membutuhkan pengaturan tambahan nantinya.



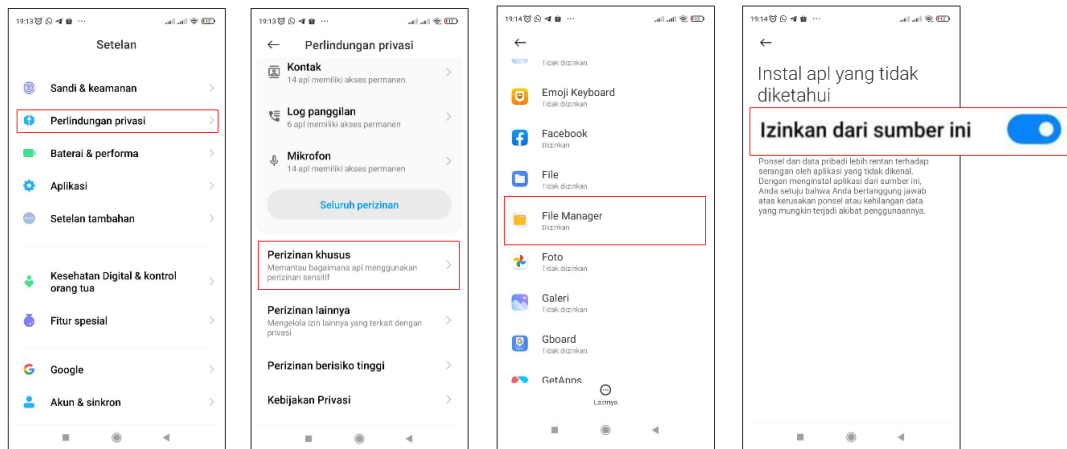
Gambar 253. proses publikasi *AIR for Android*

10.3.3 Mengujicoba *File* APK pada Gawai

Salah satu prosedur publikasi aplikasi adalah mengujicoba aplikasi yang telah dibuat terlebih dahulu secara internal. Pada tahapan ini perlu diujicoba seluruh fitur yang ada, dan memastikan seluruh fitur bekerja dengan baik. Tahapan ujicoba sebisa mungkin dilakukan pada beberapa perangkat dengan spesifikasi hardware yang berbeda untuk mengetahui performa aplikasi di masing-masing perangkat.

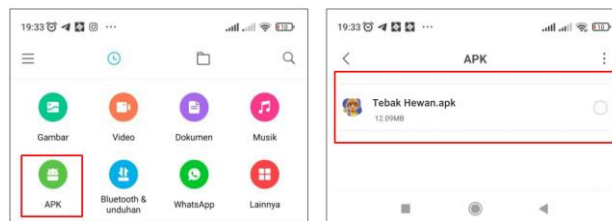
Untuk menguji coba *file* APK yang telah dibuat pada tahapan sebelumnya, *file* terlebih dahulu dipindahkan ke perangkat melalui *transfer file (via USB)*. Prosedur pemindahan *file* dari PC ke gawai Android tidak dijelaskan secara rinci, karena masing-masing gawai memiliki fitur yang berbeda-beda dalam memindah *file*. Setelah *file* berhasil dipindahkan, maka terdapat beberapa langkah yang harus dilakukan untuk menginstall aplikasi :

1. Atur *setting* pada gawai untuk memberikan izin menginstall dari sumber yang tidak diketahui. Pada umumnya pengaturan ini berada pada opsi **Perlindungan Privasi > Perizinan Khusus**. *File* APK yang dipindahkan ke gawai, pada umumnya dapat diakses oleh aplikasi **File Manager**, sehingga pada opsi File Manager, aktifkan opsi **Izinkan dari sumber ini**.



Gambar 254. mengatur perizinan instalasi aplikasi

2. Buka **File Manager**, dan pilih opsi **APK**, kemudian pilih *file* APK yang telah disimpan ke dalam gawai.



Gambar 255. memilih *file* APK

3. Ikuti proses instalasi sampai selesai, dan ujicoba seluruh fitur aplikasi.

10.3.4 Mode Layar Penuh Android

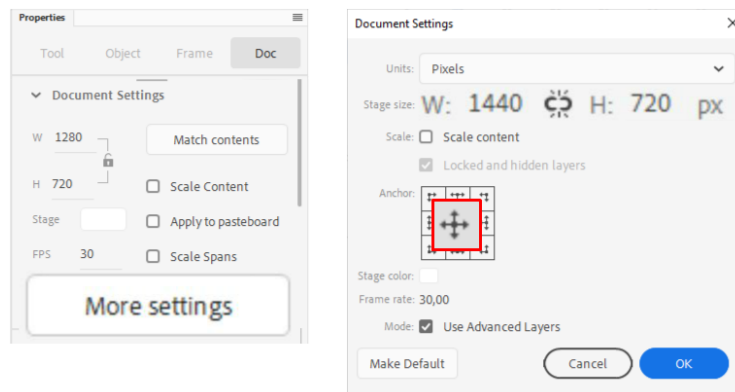
Pada beberapa gawai Android, aplikasi yang ditampilkan tidak sepenuhnya *fullscreen*, akan tetapi terdapat sedikit area di sebelah kanan dan kiri aplikasi (dengan asumsi aplikasi dengan pengaturan *landscape*/mendatar) yang berwarna hitam atau putih. Hal tersebut dikarenakan pengaturan *aspect ratio* yang kurang tepat. Sebagai contoh, aplikasi multimedia interaktif Tebak Satwa yang dibuat pada tutorial sebelumnya memiliki ukuran layar 1280x720 *pixel*, yang merupakan resolusi *High Definition (HD)* dengan *aspect ratio* 16:9. *Aspect ratio* tersebut ditujukan untuk layar komputer atau layar televisi. Sementara *aspect ratio* gawai Android khususnya *mobile phone* pada umumnya memiliki *aspect ratio* 2:1. Perbedaan tersebut akan menghasilkan area putih/hitam di sisi kanan dan kiri aplikasi.



Gambar 256. tampilan pada *mobile phone* Android

Untuk mengatasi hal tersebut terdapat beberapa langkah yang harus dilakukan, yaitu :

1. Buka kembali *file* **tutorial 8 – Tebak Satwa**. Kemudian **save as** dengan nama lain agar *file* yang digunakan untuk *publishing* dengan luaran PC tetap ada.
2. Langkah pertama untuk menyesuaikan **aspect ratio** layar Android adalah dengan mengubah ukuran proyek. Buka panel **Properties** kemudian ubah ukuran lebar dari aplikasi, karena *aspect ratio* yang diperlukan adalah 2:1, maka langkah yang paling mudah adalah mengubah ukuran lebar menjadi dua kali ukuran tinggi. Sehingga ukuran proyek perlu diubah menjadi 1440x720 *pixel*. Untuk mengubah lebar, klik opsi **More settings** pada kolom Document Setting. Selanjutnya masukkan angka 1440 pada **Width** dan pastikan **Anchor** point berada di tengah. Klik **OK** maka ukuran *Stage* akan berubah.



Gambar 257. mengubah ukuran dokumen

- Setelah ukuran *Stage* berubah, akan muncul jarak disebelah kanan dan kiri layar. Untuk mengatasi hal tersebut, klik **Frame 10 Layer background** dan ubah ukuran gambar latar. Sesuaikan dengan ukuran yang baru. Lakukan pengaturan latar pada *Keyframe* lainnya. Apabila terdapat posisi tombol yang masih dinilai kurang tepat, atur ulang dengan menggesernya.



Gambar 258. pengaturan ulang aset visual setelah perubahan ukuran *Stage*

- Simpan *file*, tekan opsi **File > Publish** untuk menghasilkan *file* APK. Selanjutnya buka aplikasi **Windows Explorer** dan akses *folder* tempat *file* proyek berada. Pada *folder* tersebut akan didapati *file* dengan nama **Tebak-Satwa-app.xml** (atau nama lain sesuai dengan nama *file* dengan tambahan **-app.xml**). *File* XML tersebut adalah sebuah *file* pengaturan aplikasi android.

Buka *file* XML tersebut dengan aplikasi *text editor*. Pada contoh ini digunakan aplikasi **Notepad++** untuk membuka *file*.

```
19 <?xml version="1.0" encoding="UTF-8" ?>
20 <application xmlns="http://ns.adobe.com/air/application/50.0">
21   <id>TebakSatwa</id>
22   <versionNumber>1.0.0</versionNumber>
23   <versionLabel/>
24   <filename>Tebak Satwa</filename>
25   <description/>
26   <name>Tebak Satwa</name>
27   <copyright/>
28   <initialWindow>
29     <content>
30       tutorial%20%20-%20IMM%20Tebak%20Satwa%20-%2018%20-%20kursor.swf
31     </content>
32     <systemChrome>standard</systemChrome>
33     <transparent>false</transparent>
34     <visible>true</visible>
35     <fullScreen>true</fullScreen>
36     <aspectRatio>landscape</aspectRatio>
37     <renderMode>cpu</renderMode>
38     <autoOrients>false</autoOrients></initialWindow>
39   <icon/>
40   <customUpdateUI>false</customUpdateUI>
41   <allowBrowserInvocation>false</allowBrowserInvocation>
42 </application>
```

Gambar 259. membuka *file* -app.XML

Pada beberapa baris terakhir perlu ditambahkan *tag* (pada contoh di buku ini, penambahan kode ditulis dengan warna merah) untuk mengatur *aspect ratio* sebagai berikut :

```
<application xmlns="http://ns.adobe.com/air/application/50.0">
  <id>TebakSatwa</id>
  ....
  <allowBrowserInvocation>false</allowBrowserInvocation>
  <android>
    <manifestAdditions>
      <![CDATA[ <manifest>
        <application
          android:hardwareAccelerated="true"> <meta-data
            android:name="android.max_aspect" android:value="3.3" />
        </application>
      </manifest> ]]>
    </manifestAdditions>
  </android>
</application>
```

Simpan kembali *file* XML tersebut.

5. Kembali ke aplikasi Adobe Animate, kemudian lakukan publikasi ulang dengan menekan tombol **File > Publish**. Setelah proses publikasi selesai, *file* bertipe APK akan terbentuk, dan dapat diinstall kembali pada gawai Android untuk menguji coba fitur *fullscreen*.

Sampai dengan tahapan ini aplikasi akan ditampilkan secara penuh pada seluruh layar, tanpa adanya jarak di sebelah kanan kiri layar. Beberapa gawai Android tertentu masih memiliki sedikit jarak, namun tidak terlalu signifikan. Jarak tersebut pada umumnya diakibatkan oleh pengaturan kamera atau pengaturan tombol UI Android (tombol *home* dan tombol *back*)

10.3.5 Publikasi Bertipe AAB (*Android App Bundle*) untuk Play Store

Google Play, juga dikenal sebagai Google Play Store merupakan layanan distribusi digital yang dioperasikan dan dikembangkan oleh Google. Play Store merupakan aplikasi resmi untuk menginstall aplikasi bersertifikat yang berjalan pada sistem operasi Android dan turunannya. Google Play juga berfungsi sebagai toko media digital, menawarkan game, musik, buku, film, program televisi dan termasuk di dalamnya aplikasi multimedia interaktif.

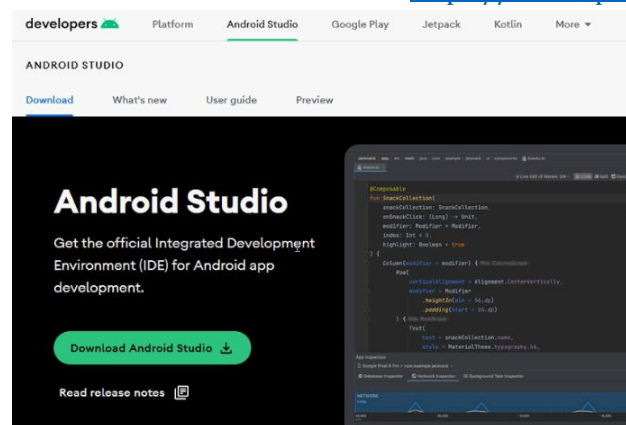
Aplikasi tersedia melalui Google Play baik secara gratis maupun berbayar. Aplikasi tersebut dapat diunduh secara langsung langsung di perangkat Android melalui aplikasi seluler Google Play Store. Google Play Store memiliki lebih dari 37 miliar unduhan aplikasi pada tahun 2022 dan lebih dari 5 juta aplikasi dipublikasikan pada tahun 2022. Hal ini menunjukkan bahwa pasar aplikasi Android melalui Play Store sangatlah besar, dan mempublikasikan aplikasi melalui *platform* tersebut memiliki jangkauan pasar yang besar.

Pada tahun 2021 kebijakan Google terbaru adalah mengubah format *file* yang dapat diunggah ke Play Store, yaitu dari *file* bertipe APK menjadi *file* bertipe AAB. *File* bertipe AAB sendiri pada dasarnya adalah satu paket *file* APK dan beberapa *file* sumber yang digunakan seperti *file* gambar, *file* suara, *file* video dan *file* terkait keamanan (sertifikat). Menggunakan *file* bertipe AAB akan mempermudah manajemen aset dan mengurangi ukuran *file* saat pengguna akan melakukan *update* terhadap aplikasi. Permasalahan yang muncul adalah, untuk mempublikasikan *file* bertipe AAB diperlukan langkah-langkah khusus.

10.3.5.1 Instalasi Android Studio

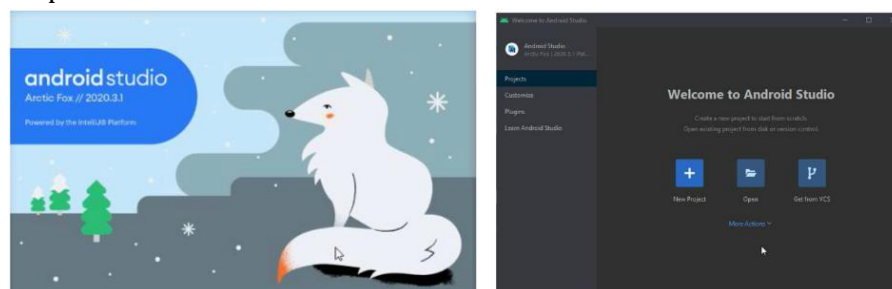
Android Studio diperlukan untuk mempublikasikan aplikasi dalam format AAB. Android Studio harus diinstall secara terpisah, dengan cara sebagai berikut :

1. **Download** *installer* Android Studio melalui situs <https://developer.android.com>



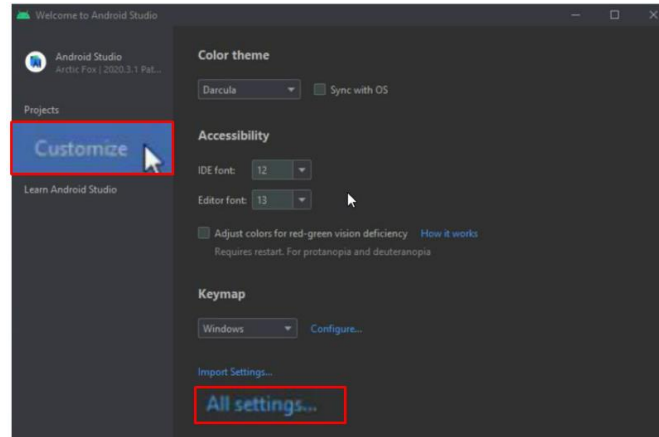
Gambar 260. halaman web Android Studio

2. Setelah *file* installer terdownload, lakukan proses instalasi aplikasi sampai selesai. Jalankan aplikasi tersebut.



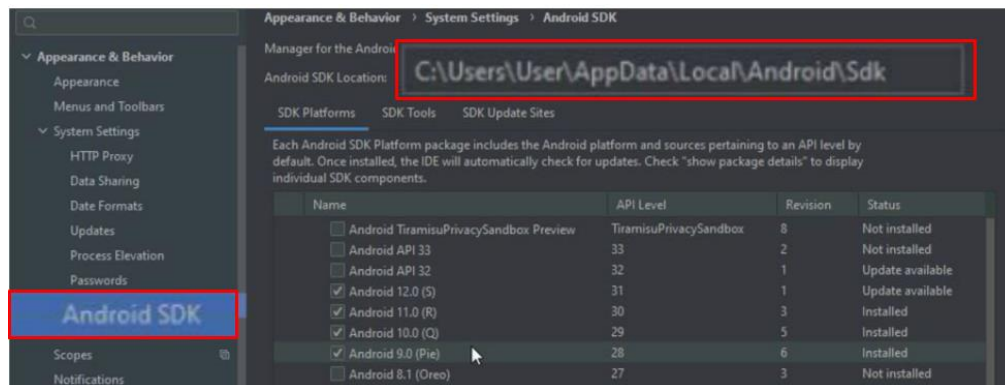
Gambar 261. menjalankan aplikasi Android Studio

3. Pada dasarnya pada aplikasi Android Studio tidak diperlukan pengaturan tertentu atau pekerjaan lain seperti pemrograman. Pada Android Studio hanya dibutuhkan lokasi tempat *SDK* Android. Klik menu **Customize**, kemudian pilih **All Settings...**



Gambar 262. halaman Setting

Selanjutnya perhatikan pada bagian **System Settings > Android SDK**. Seleksi lokasi *SDK* (perhatikan gambar), selanjutnya tekan **Ctrl+C** untuk menyalin lokasi *SDK* tersebut. Lokasi inilah yang diperlukan untuk menghasilkan *file AAB*.

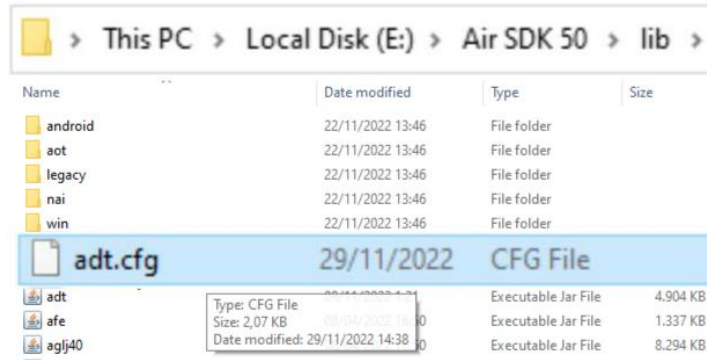


Gambar 263. lokasi Android SDK

10. 3. 5.2 Mengatur Konfigurasi AIR SDK

Setelah Android Studio terinstall, maka tahapan berikutnya adalah mengatur konfigurasi AIR SDK agar memiliki kemampuan untuk menghasilkan *file bundle*. Perhatikan langkah berikut yang merupakan kelanjutan dari langkah sebelumnya :

4. Buka *folder* instalasi/tempat mengekstrak AIR SDK (pada sub bab 10.3.1). Masuklah ke dalam *folder* **AIR SDK > lib**. Pada *folder* tersebut terdapat *file* **adt.cfg** yang merupakan *file* konfigurasi Adobe AIR. ADT merupakan singkatan dari *AIR Developer Tool* dan di dalamnya terdapat beberapa konfigurasi.



Gambar 264. lokasi *file* *adt.cfg*

5. Buka *file* tersebut dengan menggunakan aplikasi *text editor* (sebagai contoh menggunakan **Notepad++**). Di dalam *file* tersebut terdapat beberapa pengaturan yang diawali dengan tanda **#** yang berarti pengaturan tersebut **tidak aktif**. Untuk mengaktifkan pengaturan maka tanda **#** harus dihapus.

Baris pertama yang harus diaktifkan adalah baris 3 untuk mengaktifkan pengaturan perangkat.

```
DefaultArch=armv8
OverrideArch=armv8
```

Selanjutnya pada baris `CreateAndroidAppBundle` (baris 22) yang harus diset menjadi `true`. Baris ini digunakan untuk menghasilkan *file bundle*, meskipun demikian pada umumnya *file* yang dihasilkan tetap berekstensi APK dan harus diedit secara manual pada tahapan berikutnya.

```
CreateAndroidAppBundle=true
```

Baris berikutnya yang perlu diubah adalah baris 30 yaitu `path_to_sdk` dengan cara menekan **Ctrl+V** (*paste*). *Path* atau lokasi Android SDK yang *dicopy* pada langkah no 3 dipastekan ke baris tersebut.

```
AndroidPlatformSDK=C:\\Users\\User\\AppData\\Local\\Android\\Sdk
```

Perlu diperhatikan `path_to_sdk` ini menggunakan tanda `\\` (perlu ditambahkan secara manual).

Baris berikutnya yang perlu ditambahkan adalah `Java Home` (baris 34), yang dapat diisi dengan lokasi *Java Runtime* pada *folder* instalasi Android Studio

```
JAVA_HOME=C:\\Program Files\\Android\\Android Studio\\jre
```

Secara keseluruhan perubahan pada *file* **adt.cfg** adalah sebagai berikut :

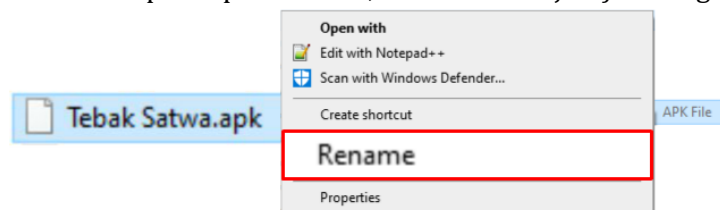
```
DefaultArch=armv8
OverrideArch=armv8
CreateAndroidAppBundle=true
AndroidPlatformSDK=C:\\Users\\User\\AppData\\Local\\Android\\Sdk
JAVA_HOME=C:\\Program Files\\Android\\Android Studio\\jre
```

6. Simpan *file* dengan menekan tombol **Ctrl+S**.

10.3.5.3 Menghasilkan *File* bertipe AAB

Langkah pengaturan Android *SDK* dan *AIR Configuration* hanya diperlukan satu kali saja, sehingga untuk pengembangan aplikasi dengan Adobe Animate selanjutnya, tidak membutuhkan pengaturan tersebut dan dapat langsung *publish*. Setelah proses pengaturan Android *SDK* dan mengatur *AIR configuration*, maka tahapan berikutnya adalah *publish* kembali *file* proyek dengan Adobe Animate, perhatikan langkah berikut :

1. Buka kembali *file* **Tutorial 8 – Tebak Satwa** yang telah diatur untuk *platform* Android (Sesuai dengan sub bab 10.3.4).
2. **Publish** ulang dengan menekan menu **File > Publish**, maka akan dihasilkan *file* bertipe APK.
3. Untuk mengubah *file* APK tersebut menjadi *file* AAB, buka **Windows Explorer**, klik kanan *file* APK kemudian pilih opsi **rename**, dan ubah menjadi *file* dengan ekstensi **.aab**



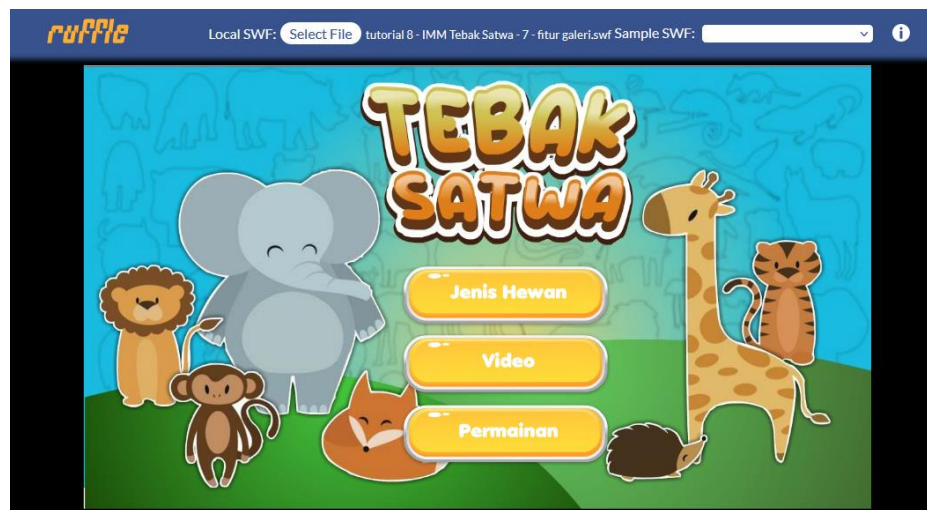
Gambar 265. *rename file* APK

4. *File* siap untuk diupload melalui Google Play Store.

10.4 Platform Web Browser

Luaran lain yang dapat dihasilkan oleh Adobe Animate adalah aplikasi *web browser*. Pada awal tahun 2021 Adobe secara resmi menghentikan *Flash Player*, sebuah *plugin* yang diperlukan untuk menjalankan *file* SWF pada *platform web browser*. Demikian halnya dengan beberapa *browser* populer seperti Google Chrome, Fire Fox dan Edge yang menghentikan *plugin* *Flash Player* pada tahun 2021, sehingga *web browser* tidak dapat lagi menjalankan *file* bertipe SWF. Hal ini dikarenakan fitur HTML 5 dengan dukungan Canvas dan Javascript yang semakin matang untuk menampilkan fitur interaktivitas pada *web browser*.

Namun demikian terdapat opsi untuk menjalankan *file* SWF melalui *web browser*, yaitu dengan menggunakan Ruffle. Ruffle adalah emulator *Flash Player* yang ditulis dalam bahasa Rust. Ruffle berjalan secara native di semua sistem operasi modern sebagai aplikasi mandiri, dan di semua *browser* modern melalui penggunaan *WebAssembly*. Ruffle dirancang agar mudah digunakan dan diinstal, pengguna atau pemilik situs *web* dapat menginstall versi *web* Ruffle dan konten Flash yang ada (*file* SWF) akan bekerja, tanpa memerlukan konfigurasi tambahan. Ruffle akan mendeteksi semua konten Flash yang ada di situs *web* dan secara otomatis melakukan "*polyfill*" ke dalam sebuah konten berbasis HTML 5 yang memungkinkan pemutakhiran situs *web* yang lancar.



Gambar 266. tampilan multimedia interaktif pada *web browser*

Untuk menjalankan aplikasi pada *web browser*, terdapat beberapa langkah yang harus dilakukan, yaitu :

1. Buka kembali *file* proyek **tutorial 8 - Tebak Satwa** (versi PC) dan pilih menu **File > Publish** untuk menghasilkan *file* bertipe **SWF** (pada contoh ini dihasilkan *file* dengan nama Tebak Satwa.swf).
2. Buka aplikasi teks editor seperti **Notepad++**, kemudian *copy-paste* kode berikut :

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">

<html xmlns="http://www.w3.org/1999/xhtml" lang="pt" xml:lang="en">
  <head>
    <title></title>
    <meta http-equiv="Content-Type" content="text/html;
    charset=utf-8" />
    <style type="text/css" media="screen">

    html, body { height:100%; background-color: #ffffff;
```



```

leftmargin="0"; topmargin="0"; marginwidth="0";
marginheight="0"; }

:root {
--preloader-background: #FFFFFF;
--logo-display: block;

}

body { margin:0; padding:0; overflow:hidden; }
#flashContent { width:100%; height:100%; }

</style>

<!--[Start my Ruffle]>-->

<script src="ruffle.js"></script>
<script>
window.RufflePLayer = window.RufflePLayer || {};
window.RufflePLayer.config = {
// Options affecting files only
"autoplay": "on",
"letterbox": "on",
"unmuteOverlay": "hidden",
"warnOnUnsupportedContent": false,
"contextMenu": false,
"upgradeToHttps": window.location.protocol === "https:",
};
</script>

<!--[End my Ruffle]>-->

</head>
<body>
<div id="flashContent">
<object classid="clsid:id" width="100%" height="100%"
id="Tebak Satwa" align="middle">
<param name="movie" value="Tebak Satwa.swf" />
<param name="quality" value="high" />
<param name="bgcolor" value="#ffffff" />
<param name="play" value="true" />
<param name="loop" value="true" />
<param name="wmode" value="window" />
<param name="scale" value="showall" />
<param name="menu" value="true" />
<param name="devicefont" value="false" />
<param name="salign" value="" />
<param name="allowScriptAccess" value="sameDomain" />
</object>
</div>
</body>
</html>

```

3. Simpan dengan nama **TebakSatwa.html**, pastikan *file* tersebut berada satu *folder* dengan *file* SWF.
4. **Upload** *file* HTML dan *file* SWF ke *web server*. Pengujian menggunakan Ruffle harus dilakukan pada *web server (online)* atau setidaknya dilakukan pengujian dengan menggunakan pengaturan *Localhost*.

Mengupload ke *server* berarti meletakkan seluruh *file* yang dibutuhkan pada sebuah *web hosting* dengan alamat domain tertentu. Proses ini dapat dilakukan dengan menggunakan aplikasi FTP seperti *FileZilla*. Setelah *file terupload* ke sebuah *server* maka dapat diakses melalui alamat web **namaDomain.com/TebakSatwa.html**.

Apabila masih belum memiliki *server*, ujicoba juga dapat dilakukan secara lokal atau diistilahkan dengan *localhost*. *Localhost* adalah pengaturan untuk menjadikan komputer sebagai *server* virtual yang aksesnya terbatas pada jaringan lokal/*offline*. *Localhost* berperan sebagai domain dari *server* lokal dengan alamat IP-nya yaitu 127.0.0.1. Untuk mengakses *localhost* bisa dengan menuliskan `http://localhost` atau 127.0.0.1 pada *internet browser*. Untuk mempermudah penggunaan *localhost* pada umumnya digunakan aplikasi local *server*, salah satunya adalah XAMPP.

BAB 11

Penutup

Keunggulan utama multimedia interaktif adalah interaktivitas yang ada di dalam aplikasi dapat membuka berbagai peluang interaksi antara pengguna dengan media. Kendati demikian untuk membentuk interaktivitas yang baik diperlukan pengetahuan yang baik tentang desain antar muka dan teknik pemrograman, dua hal yang menjadi kendala bagi sebagian besar pengembang. Melalui buku ini, penulis mencoba mendeskripsikan proses pembuatan multimedia interaktif secara komprehensif mulai dari perencanaan, desain hingga teknik pemrograman.

Pengembangan multimedia interaktif menggunakan aplikasi Adobe Animate pada dasarnya memiliki proses yang relatif sederhana, dimulai dari perencanaan, pengembangan aset visual, pemrograman dan proses publikasi. Dalam proses pemrograman, *MPI Component* dapat digunakan untuk mempermudah pengkodean. Pengkodean yang sebelumnya relatif kompleks membutuhkan beberapa baris untuk menjalankan sebuah fungsi tertentu, dapat diringkas dengan pemanfaatan *compiled clip*. Pada tahapan *publishing* Adobe Animate dapat menghasilkan luaran aplikasi *multiplatform* yang dapat bekerja pada beberapa *platform* seperti PC, Android dan *Web browser*.

Sebagai penutup, semoga buku ini memberikan manfaat kepada pembaca dan memberikan manfaat bagi pendidikan di Indonesia. Penulis berterimakasih kepada seluruh pembaca yang telah meluangkan waktu untuk menambah sedikit pengetahuan terkait pengembangan aplikasi multimedia interaktif melalui buku ini. Secara khusus ucapan terimakasih juga penulis sampaikan kepada para mahasiswa DKV Universitas Negeri Semarang pada mata kuliah multimedia interaktif yang selama ini telah menghasilkan karya-karya terbaik baik untuk kepentingan portfolio personal, proyek studi maupun untuk kepentingan komersial. Besar harapan penulis agar sedikit pengetahuan yang disampaikan melalui buku ini menjadi sebuah ilmu yang bermanfaat yang dapat digunakan untuk mengembangkan aplikasi yang juga bermanfaat bagi khalayak.

DAFTAR PUSTAKA

- Abdullah, D. H. (2021). Kuliner Khas Nusantara. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Ahmad, Z. (2021). Batik Kita. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Akbar, Y. (2021). *Know Your Camera*. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Amanata, K. (2022). *Bioma*. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Amin, M.H. (2021). Sinau Batik. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Arzak, N. (2021). *Vegefruit*. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Azahra, F. (2018). *Scuba Diving*. tangkap layar aplikasi Multimedia Interaktif. D3 Desain Komunikasi Visual UNNES.
- Borg, W.R. & Gall, M.D. Gall. (1983). *Educational Research: An Introduction*,. Fifth Edition. New York: Longman
- Falikh, B. (2020). Mengenal Tokoh Uang. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Firdaus, A. (2021). Mengenal P3K. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Firly, C. (2022). *I am Mammals*. tangkap layar aplikasi Multimedia Interaktif. D3 Desain Komunikasi Visual UNNES.
- Halawah, H. (2022). Cara menjadi Youtuber. tangkap layar aplikasi Multimedia Interaktif. D3 Desain Komunikasi Visual UNNES.
- Kemenperin. (2016). SKKNI Desain Grafis/Desain Komunikasi Visual.
- Lidya, M. (2021). Bersih-Bersih Yuk! . tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Oktavian, I. A. (2022). Laboran. Dokumen Desain Naskah Game Edukasi. Semarang: BPTIKP Dikbud Jawa Tengah.
- Oktaviardiani, S.F.B. (2022). Mengenal Lapisan Bumi. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Murtiningsih. (2013). Pakaian dan Rumah Adat Nusantara. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Mustaqim, A. (2021). *Inside Our Body*. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Nisa, R. K. (2022). Mengenal Rempah Nusantara. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.

- Novrizal, A. (2020). Pahlawan Nasional Indonesia. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Priyambodo. (2021). Ayo Belajar Tata Surya!. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Putri, W. (2022). 4 Sehat 5 Sempurna. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Rizka, A. (2022). *Fire Fight*. tangkap layar aplikasi Multimedia Interaktif. D3 Desain Komunikasi Visual UNNES.
- Rosalind, B. (2021). *Piliana Cultourism*. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Sabrina, P. (2022). *Ngigel*. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Saad, K. (2021). Satwa Endemik Indonesia. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Salsabila, T. (2021). Menjelajah Bumi Raflesia. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Santoso, B. K. (2022). Tur Planet. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Satria, F. (2020). Mari Belajar Membuat Komik. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Setyaningrum. D. (2022). Ayo Memilah Sampah. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Setyaningrum, L. (2021). Lincih Matika. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Sivana, M. (2022). Bendera Dunia. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Ulqalbi, E. A. (2021). *Flower Garden*. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Wandayanti, M. (2022). Cerita Affan. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Wibawanto, W. (2017). Desain dan Pemrograman Multimedia Pembelajaran Interaktif. Jember: Cerdas Ulet Kreatif.
- Wibawanto, W. (2020). Laboratorium Virtual : Konsep dan Pengembangan Simulasi Fisika. Semarang: Penerbit LPPM UNNES
- Wibawanto, W. (2014). *Astro Farmer*. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.
- Wibawanto, W. dan Widagdo, P. B. (2021). *Onsertify*. tangkap layar aplikasi www.onsertify.com
- Zuhri, F. (2020). Panduan Umrah. tangkap layar aplikasi Multimedia Interaktif. S1 Desain Komunikasi Visual UNNES.

TENTANG PENULIS

Dr. Wandah Wibawanto S.Sn., M.Ds. secara khusus mendalami pengembangan game dan aplikasi sejak tahun 2001. Menyelesaikan program S1 Desain Komunikasi Visual di Universitas Negeri Malang pada tahun 2006, S2 Magister Desain (Game Tech/Media Digital) ITB pada tahun 2012 dan menyelesaikan program S3 Pendidikan Seni Universitas Negeri Semarang pada tahun 2022

Pada tahun 2006 bekerja di sebuah perusahaan game FreeOnline Games.com dan menghasilkan beberapa karya game yang cukup populer, di antaranya adalah Sim Taxi, Gangster Life, The Empire, dan beberapa game lainnya. Tahun 2008 memutuskan untuk mengembangkan game flash secara independen untuk beberapa game portal seperti Armorgames, Gameninja, Gamebooks, DailyFreeGames, dan Froyogames.com. Mendapatkan beberapa penghargaan di bidang aplikasi dan games seperti pemenang INAICTA 2009 kategori digital media, Honorable mention Dictionary Games award, FOG daily game programing challenge dan beberapa lainnya.

Tahun 2014 berpindah ke bidang akademik dengan bergabung menjadi dosen DKV Universitas Negeri Semarang, menjadi peneliti di bidang ekonomi kreatif, dan pengembangan media edukasi. Beberapa buku yang telah ditulis terkait dengan pengembangan Aplikasi antara lain :

- Membuat Game dengan Flash (2003) Penerbit Andi, Yogya
- Dasar-dasar Pemrograman Flash Game (2006) e-book
- Membuat Flash Game 3D (2013) Penerbit Andi, Yogya
- Desain dan Pemrograman Multimedia Interaktif (2017) Penerbit Cerdas Ulet Kreatif, Jember
- Membuat bermacam-macam Game Android dengan Adobe Animate (2019) Penerbit Andi, Yogya
- Game Edukatif RPG (2020), Penerbit LPPM UNNES, Semarang
- Laboratorium Virtual (2020), Penerbit LPPM UNNES, Semarang

Beberapa buku terkait ekonomi kreatif antara lain :

- Book chapter "Kolase Pemikiran Ekonomi Kreatif" (2017)
- Pendampingan OPD dalam Pengembangan Ekonomi Kreatif Kab/Kota (2018)
- Integrasi Rencana Induk Pengembangan Ekonomi Kreatif (Rindekraf) dalam RPJMD Kab/Kota (2019)

Beberapa karya aplikasi, tutorial dan materi perkuliahan dapat ditemukan di situs wandah.com, wandah.org dan channel youtube www.youtube.com/user/wandahw. Pertanyaan terkait pengembangan aplikasi multimedia atau yang lain dapat disampaikan melalui email wandah@wandah.com atau wandah@mail.unnes.ac.id

Teknik Cepat Pengembangan MULTIMEDIA INTERAKTIF

dengan **MPI Component**

Keunggulan utama multimedia interaktif adalah interaktivitas yang ada di dalam aplikasi dapat membuka berbagai peluang interaksi antara pengguna dengan media. Kendati demikian untuk membentuk interaktivitas yang baik diperlukan pengetahuan yang baik tentang desain antar muka dan teknik pemrograman, dua hal yang menjadi kendala bagi sebagian besar pengembang.

Melalui buku ini, penulis mencoba mendeskripsikan proses pembuatan multimedia interaktif secara komprehensif mulai dari perencanaan, desain hingga teknik pemrograman. Pengembangan multimedia interaktif menggunakan aplikasi Adobe Animate pada dasarnya memiliki proses yang relatif sederhana, dimulai dari perencanaan, pengembangan aset visual, pemrograman dan proses publikasi.

Dalam proses pemrograman, MPI Component dapat digunakan untuk mempermudah pengkodean. Pengkodean yang sebelumnya relatif kompleks membutuhkan beberapa baris untuk menjalankan sebuah fungsi tertentu, dapat diringkas dengan pemanfaatan compiled clip.

Buku ini membahas secara detail penggunaan MPI Component untuk membangun aplikasi multimedia dengan cepat, mudah dan efektif. Buku ini dapat digunakan oleh pembelajar pemula, guru, mahasiswa maupun pengembang aplikasi yang menggunakan Adobe Animate dalam mengembangkan produknya.



Hak Cipta © pada Penulis dan dilindungi Undang-undang Penerbitan.
Hak Penerbitan pada Unnes Press | Dicitak oleh Unnes Press
Jl. Kelud Raya No. 2 Semarang 50237 | Telp. (024) 86008700 ext. 062



ISBN 978-602-285-378-7



9

786022

853787