

Nur Aini Rakhmawati

MAMA ANDROID



**PRODISTIK
ITS**

Mama Android

Penulis : Nur Aini Rakhmawati
Desain Sampul : Somantha Adyprima Nigraha

Hak Cipta dilindungi undang-undang.

Dilarang memperbanyak buku ini dalam bentuk apapun tanpa seijin penulis dan penerbit.

Diterbitkan oleh:,
Prodistik ITS
Surabaya

Katalog Dalam Terbitan (KDT)
Nur Aini Rakhmawati, Mama Android—Cetakan 1 – Surabaya:
Prodistik ITS, 2016
xii + 100; 14 x 20cm

ISBN: XXXX

Cetakan 1, 2016

KATA PENGANTAR

Dengan menyebut nama Allah SWT yang Maha Pengasih lagi Maha Penyayang, kami panjatkan puja dan puji syukur atas kehadiran-Nya, yang telah melimpahkan rahmat kepada kami, sehingga kami dapat menyelesaikan buku ajar pemrogramaan berbasis android dengan menggunakan App Inventor.

Terlepas dari semua itu, kami menyadari sepenuhnya bahwa masih ada kekurangan baik dari segi susunan kalimat maupun tata bahasanya. Oleh karena itu dengan tangan terbuka kami menerima segala saran dan kritik dari pembaca.

Akhir kata kami berharap semoga modul ini bermanfaat bagi mahasiswa dan pelajar maupun kalangan umum yang ingin belajar Android.

Surabaya, 4 Juni 2016

Penulis

*Seekor induk lumba-lumba beratraksi mempesona penonton
yang melihatnya.*

*Tiba-tiba datanglah anak lumba-lumba. Merekapun berduet
atraksi dengan indahnya.*

*Atraksi ini adalah atraksi yang sama dilakoni oleh induk
lumba-lumba tersebut ketika belum mempunyai anak.*

*Kehadiran anak lumba-lumba lebih mempercantik atraksi
tersebut*

*Tidak ada kata berhenti berkarya ketika titel ibu kita sandang.
Bahkan kehadiran seorang anak memberikan energi yang luar
biasa untuk menjadi lebih kreatif dalam berkarya di kehidupan.*

Mama - Key - Naz

DAFTAR ISI

Kata Pengantar	i
DAFTAR ISI	iii
DAFTAR GAMBAR	vi
DAFTAR TABEL	ix
1 Si Manis Android	1
1.1 Tujuan	1
1.2 3 Alasan mengapa mengembangkan aplikasi Android	1
1.3 App Inventor	2
1.3.1 Android Studio Vs App Inventor	2
1.3.2 Komponen App Inventor	3
1.4 Instalasi App Inventor	4
1.4.1 Online	5
1.4.2 Lokal	5
1.5 Menjalankan Aplikasi Android	6
1.6 Berbagi Aplikasi	8
1.7 Pemaketan Aplikasi	10
2 Your name is ...	11
2.1 Tujuan	11
2.2 Pembuatan Program	11
2.2.1 Tampilkan Namamu	11
2.2.2 Katakan Namamu	13
3 Senyum dan Klik	16
3.1 Tujuan	16
3.2 Pembuatan Program	16
3.2.1 Screen: Ambil Foto	17
3.2.2 Screen: Rekam Video	19

4	Phantom Vibrations Syndrome	22
4.1	Tujuan	22
4.2	Layout Pada App Inventor	22
4.3	Accelerometer Sensor	23
4.4	Variabel Global	25
4.5	Pembuatan Program	25
5	Merah Kuning Hijau	31
5.1	Tujuan	31
5.2	Pembuatan Program	31
6	Di sini aku berada !	35
6.1	Tujuan	35
6.2	GPS	35
6.3	Pembuatan Program	35
7	Lemparkan Bola itu	39
7.1	Tujuan	39
7.2	Pembuatan Program	39
8	Simpan datamu	42
8.1	Tujuan	42
8.2	TinyDB	42
8.3	Variabel Lokal	43
8.4	Pembuatan Program	43
9	Rumus	47
9.1	Tujuan	47
9.2	Procedure	47
9.3	Pembuatan Program	49
10	Kirimkan pesanmu	53
10.1	Tujuan	53
10.2	List	53
10.3	Pembuatan Program	54

11 Mari berhitung	58
11.1 Tujuan	58
11.2 IF ELSE	58
11.3 FOR EACH	60
11.4 Pembuatan Program	61
12 Baca dan Tulis	64
12.1 Tujuan	64
12.2 File	64
12.3 Pembuatan Program	65
13 Sotonya Enak	69
13.1 Tujuan	69
13.2 Pembuatan Program	69
14 Denting Waktu	73
14.1 Tujuan	73
14.2 Clock	73
14.3 Pembuatan Program	74
DAFTAR PUSTAKA	78
INDEKS	81

DAFTAR GAMBAR

1.1	Designer	4
1.2	Blocks	4
1.3	run	6
1.4	QR Code	7
1.5	MIT AI2 Companion	7
1.6	Emulator	9
1.7	Berbagi Aplikasi	9
1.8	APK	10
2.1	Komponen Tampilkan Namamu pada App In- ventor	12
2.2	Desain Aplikasi Tampilkan Namamu	13
2.3	Blocks Editor Aplikasi Tampilkan Namamu	13
2.4	Tampilan Aplikasi Tampilkan Namamu	14
2.5	Komponen Katakan Namamu pada App Inventor	14
2.6	Blocks Editor Aplikasi Katakan Namamu	15
3.1	Desain Komponen Senyum di Screen 1	17
3.2	Desain Aplikasi Senyum di Screen 1	18
3.3	Blocks Editor Aplikasi Senyum di Screen 1	19
3.4	Tampilan Aplikasi Senyum di Screen 1	20
3.5	Desain Komponen Senyum di Screen 2	20
3.6	Desain Aplikasi Senyum di Screen 2	20
3.7	Blocks Editor Aplikasi Senyum di Screen 2	21
3.8	Tampilan Aplikasi Senyum di Screen 2	21
4.1	Jenis Layout pada App Inventor	23
4.2	Tampilan Tiga Jenis Layout	24
4.3	Accelerometer pada smartphone [1]	24
4.4	Deklarasi variabel global	25
4.5	Komponen GetarXYZ pada App Inventor	26
4.6	Blocks Editor Aplikasi GetarXYZ	27
4.7	Penggabungan teks	28
4.8	Tampilan Aplikasi GetarXYZ	29
4.9	Komponen Vibrasi pada App Inventor	29

4.10	Desain Aplikasi vibrasi	29
4.11	Blocks Editor Aplikasi Vibrasi	30
5.1	Komponen aplikasi Gambar pada App Inventor .	32
5.2	Desain Aplikasi Gambar	33
5.3	Mengambil Parameter	33
5.4	Blocks Editor Aplikasi Gambar	34
5.5	Tampilan Aplikasi Warna	34
6.1	Komponen aplikasi GPS pada App Inventor . . .	36
6.2	Desain Aplikasi GPS	37
6.3	Blocks Editor Aplikasi GPS	37
6.4	Tampilan Aplikasi GPS	38
7.1	Komponen aplikasi Bola pada App Inventor . . .	40
7.2	Desain Aplikasi Bola	40
7.3	Blocks Editor Aplikasi Bola	41
7.4	Tampilan Aplikasi Lokasi	41
8.1	Fungsi TinyDB	43
8.2	Contoh Variable Lokal	44
8.3	Komponen aplikasi Kontak pada App Inventor .	45
8.4	Desain Aplikasi Kontak	45
8.5	Blocks Editor Aplikasi Kontak	46
8.6	Tampilan Aplikasi Kontak	46
9.1	<i>procedure</i> tanpa parameter dan pengembalian . .	48
9.2	Parameter dari sebuah <i>procedure</i>	48
9.3	<i>procedure</i> dengan parameter dan tanpa pengem- balian	48
9.4	<i>procedure</i> dengan parameter dan pengembalian .	49
9.5	Komponen aplikasi Luas pada App Inventor . . .	50
9.6	Desain Aplikasi Luas	50
9.7	Blocks Editor Aplikasi Luas 1	51
9.8	Blocks Editor Aplikasi Luas 2	51
9.9	Blocks Editor Aplikasi Luas 3	51
9.10	Tampilan Aplikasi Luas	52

10.1	Contoh Membuat List Baru	54
10.2	Contoh Menambah data List	54
10.3	Contoh Mengakses data di List	55
10.4	Komponen aplikasi SMS pada App Inventor . . .	55
10.5	Desain Aplikasi SMS	56
10.6	Blocks Editor Aplikasi SMS	56
10.7	Tampilan Aplikasi SMS	57
11.1	IF	59
11.2	IF ELSE	59
11.3	FOR EACH	60
11.4	FOR EACH dan LIST	61
11.5	Komponen aplikasi Hitung pada App Inventor .	62
11.6	Desain Aplikasi Hitung	62
11.7	Blocks Editor Aplikasi Hitung	63
11.8	Tampilan Aplikasi Hitung	63
12.1	Operasi File pada App Inventor	64
12.2	Komponen aplikasi data siswa pada App Inventor	66
12.3	Desain Aplikasi Data Siswa	67
12.4	Blocks Editor Aplikasi Data Siswa	67
12.5	Tampilan Aplikasi Data Siswa	68
13.1	Komponen aplikasi Soto pada App Inventor . . .	70
13.2	Desain Aplikasi Soto	70
13.3	Blocks Editor Aplikasi Soto	71
13.4	Tampilan Aplikasi Soto	72
14.1	Komponen aplikasi Waktu pada App Inventor . .	75
14.2	Desain Aplikasi Waktu	75
14.3	Blocks Editor Aplikasi Waktu	76
14.4	Tampilan Aplikasi Waktu	77

DAFTAR TABEL

1.1	Perbandingan Android Studio dan App Inventor	3
2.1	Daftar Komponen Tampilkan Namamu	12
2.2	Daftar Komponen Katakan Namamu	14
3.1	Daftar Komponen Aplikasi Senyum	17
4.1	Daftar Komponen Aplikasi Bergoyang	26
5.1	Daftar Komponen Aplikasi Gambar	32
6.1	Daftar Komponen Aplikasi GPS	36
7.1	Daftar Komponen Aplikasi Bola	39
8.1	Daftar Komponen Aplikasi Simpan	44
9.1	Daftar Komponen Aplikasi Luas	49
10.1	Daftar Komponen Aplikasi SMS	54
11.1	Daftar Komponen Aplikasi Hitung	61
12.1	Daftar Komponen Aplikasi Data Siswa	65
13.1	Daftar Komponen Aplikasi Soto	69
14.1	Daftar Komponen Aplikasi Waktu	74

BAB 1

SI MANIS ANDROID

Apakah kamu suka makanan yang manis-manis, seperti permen dan es krim? Tahukah kamu, Android dirilis dengan nama makanan yang manis dengan urutan sebagai berikut[2]

Cupcake - Donut - Eclair - Froyo - Gingerbread - Honeycomb - Ice cream sandwich - Jelly bean - Kitkat - Lollipop - Marshmallow

Kali ini, kita akan memulai mengenalkan Android sebagai sistem operasi yang digunakan pada smartphone dan alasan mengapa kita mengembangkan aplikasi Android. App Inventor sebagai salah satu perangkat untuk mengembangkan Android juga akan dibahas di bab ini.

1.1 Tujuan

1. Mengetahui komponen utama dari App Inventor
2. Menggunakan App Inventor secara online
3. Melakukan instalasi App Inventor di lokal komputer
4. Melakukan download aplikasi yang sudah dibuat

1.2 3 Alasan mengapa mengembangkan aplikasi Android

Mengapa mengembangkan aplikasi android sangat menguntungkan daripada aplikasi mobile lainnya? Yuk kita simak tiga alasan utama berikut ini:

1. *Market Share* Android di Indonesia cukup besar.

Munurut StatCounter Global Stats [3], lebih dari 70% orang Indonesia menggunakan Android sebagai sistem operasi di smartphone.

2. Keuntungan yang menjanjikan

Sebuah perusahaan pemasaran game bernama DAU-UP [4] menyatakan bahwa rata-rata keuntungan yang didapatkan dari setiap user pada aplikasi android adalah 20% diatas iOS.

3. Dapat dikembangkan dimanapun

Android dikembangkan dengan bahasa *Java* yang dapat dikerjakan pada berbagai sistem operasi. Berbeda dengan aplikasi yang berjalan di iOS harus dikembangkan di atas Mac.

1.3 App Inventor

App Inventor (<http://appinventor.mit.edu/>) merupakan sebuah software untuk mengembangkan Android dengan cara *drag and drop* [5]. Dengan App Inventor, kita tidak perlu menuliskan koding untuk menciptakan sebuah aplikasi Android. Aplikasi ini disusun seperti *puzzle*. Software ini awalnya di kembangkan oleh Google dan dikembangkan lebih lanjut oleh MIT.

1.3.1 Android Studio Vs App Inventor

Android Studio (<https://developer.android.com/studio/index.html>) merupakan aplikasi resmi dari Google yang dipakai untuk mengembangkan aplikasi Android. Pada Tabel 1.1, kita membandingkan Android Studio dengan App Inventor.

Tabel 1.1: Perbandingan Android Studio dan App Inventor

Komponen	Android Studio	App Inventor
Kecepatan	Lambat	Cepat
Spesifikasi Hardware	Tinggi	Sedang
Skala Aplikasi	Menengah dan Besar	Kecil
Pengguna	Mahir	Pemula
Pengerjaan	Lokal Komputer	Cloud dan Lokal

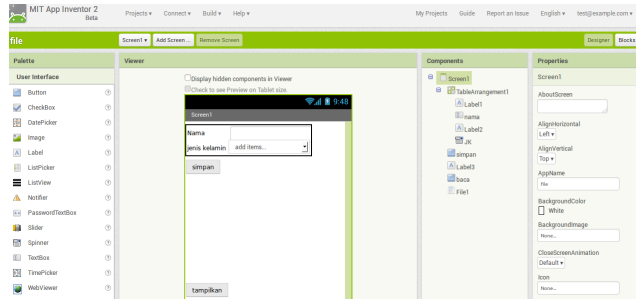
Android studio menggunakan komponen *Gradle* yang terkadang menyebabkan proses pembangunan aplikasi cukup lambat. Oleh karena itu, Android Studio membutuhkan hardware dengan spesifikasi lebih tinggi daripada App Inventor. Android Studio lebih mampu untuk digunakan dalam mengembangkan aplikasi yang berskala besar dan menengah. App Inventor sangat cocok bagi mereka yang tidak mempunyai pengetahuan tentang pemrograman. Untuk dapat menggunakan Android Studio kita perlu menginstallnya pada komputer lokal. Pengembangan aplikasi Android dengan App dapat dijalankan di cloud dengan browser atau kita dapat menginstalasi App Inventor di lokal komputer kita.

1.3.2 Komponen App Inventor

Ada dua buah komponen utama pada App Inventor yaitu *Component Designer* dan *Blocks Editor*

1. *Component Designer*

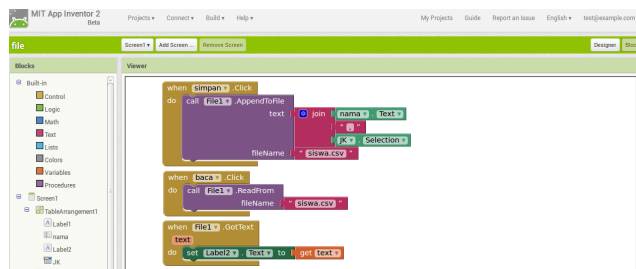
Pada *Component Designer* (Gambar 1.1) kita bisa memilih komponen apa yang akan kita pakai pada aplikasi. Komponen yang kita butuhkan cukup ditarik di bagian interface. Properti komponen tersebut dapat kita ubah di bagian paling kanan.



Gambar 1.1: Designer

2. Blocks Editor

Komponen yang sudah kita pilih perlu didefinisikan aksi dari setiap komponen tersebut pada *Blocks Editor* (Gambar 1.2). Misalnya apakah yang terjadi jika sebuah tombol dipilih, jika gambar disentuh dan lain-lain.



Gambar 1.2: Blocks

1.4 Instalasi App Inventor

Sebagaimana disebutkan pada Bab 1.3.1, kita dapat menjalankan App Inventor secara online via browser (baca: *Cloud*) atau di lokal komputer kita.

1.4.1 Online

Tidak perlu melakukan instalasi macam-macam di komputer kita. Kita bisa langsung membuat aplikasi Android dengan App Inventor di situs <http://ai2.appinventor.mit.edu/>. Cukup melakukan pendaftaran dengan *Google Account* kita. Semua aplikasi kita akan tersimpan di web tersebut.

1.4.2 Lokal

Kelemahan mengembangkan aplikasi Android di <http://ai2.appinventor.mit.edu/> adalah kita harus selalu terhubung dengan internet. Untuk mengatasi hal tersebut, kita juga dapat menginstall di komputer kita sendiri. Untuk menginstall App Inventor, kita perlu menginstall Java Development Kit (JDK) terlebih dahulu karena App Inventor merupakan aplikasi berbasis Java. JDK dapat diunduh dari <http://www.oracle.com/technetwork/java/javase/downloads/index.html>. Pada umumnya JDK sudah terinstal di komputer. Yang perlu dilakukan adalah melakukan konfigurasi *JAVA_HOME* agar sistem dapat mengetahui lokasi JDK berada.

Aplikasi App Inventor bisa diunduh di <https://sourceforge.net/projects/ailivecomplete> dalam bentuk *zip*. Setelah diunduh, kita harus mengekstrak file *zip* tersebut. Pada direktori dari hasil ekstrak kita akan mendapatkan program untuk menjalankan App Inventor.

Untuk Windows, kita perlu menjalankan file

```
WinStartAIServer.cmd
```

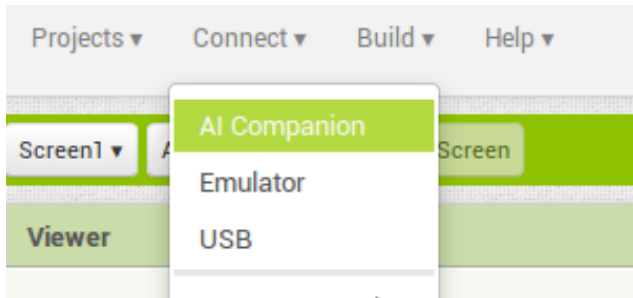
Untuk Linux, kita perlu menjalankan file

```
./startAIServer.sh
```

Setelah mengesekusi file tersebut diatas, kita bisa mulai menggunakan App Inventor dengan membuka browser dengan alamat `http://localhost:8888`.

1.5 Menjalankan Aplikasi Android

Untuk menjalankan aplikasi Android kita cukup memilih menu *Connect* (Gambar 1.3). Ada tiga cara menjalankan aplikasi Android yang kita kembangkan pada App Inventor, yaitu:

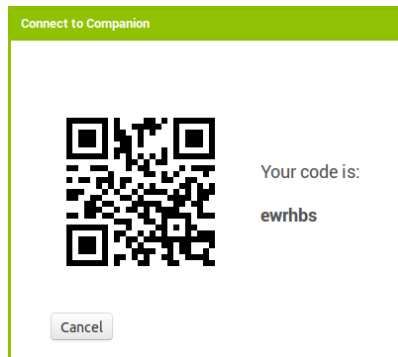


Gambar 1.3: run

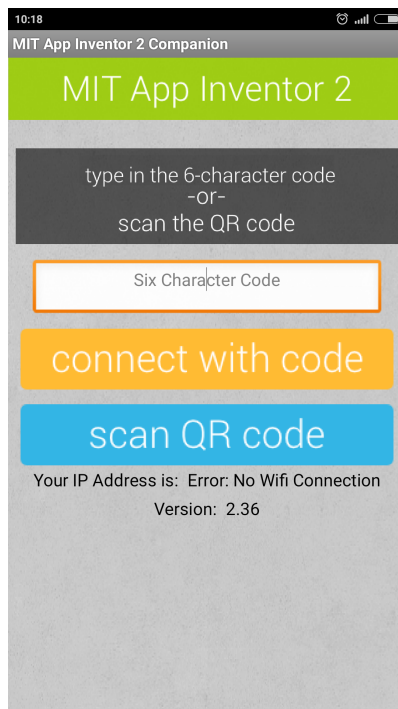
1. Smartphone + Wifi

Yang paling menarik dari App Inventor adalah kita dapat langsung menjalankan aplikasi Android pada smartphone kita. Namun kita perlu menginstal *MIT AI2 Companion* yang tersedia pada *Google Play Store*. Buka aplikasi *MIT AI2 Companion* sebelum anda memilih menu *Connect -> AI Companion*. Agar smartphone anda dapat terhubung dengan aplikasi di App Inventor maka smartphone anda harus terhubung dengan jaringan internet. Ketika QR Code aplikasi muncul

2. Smartphone + USB



Gambar 1.4: QR Code



Gambar 1.5: MIT AI2 Companion

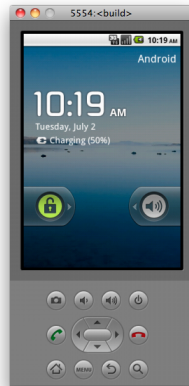
Jika tidak punya koneksi internet kita bisa menghubungkan App Inventor dengan smartphone via USB. Namun kita perlu menginstall *aiStarter* di komputer kita. *aiStarter* dapat diunduh di <http://appinventor.mit.edu/explore/ai2/setup.html>. Sebelum menghubungkan App Inventor dengan smartphone kita harus memastikan smartphone telah terkoneksi dengan baik dengan USB. Kemudian aktifkan mode USB Debugging pada smartphone yang terdapat pada *Settings > Applications > Development*. Setelah itu, *aiStarter* harus sudah berjalan di komputer kita. Kemudian jalankan MIT AI2 Companion di smartphone, maka aplikasi kita akan terlihat di smartphone

3. Emulator

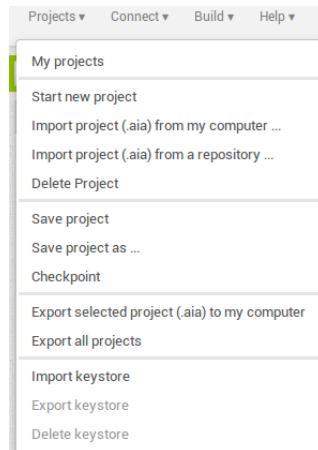
Tidak punya smartphone berbasis Android? Jangan khawatir kita tetap bisa menjalankan aplikasi kita di App Inventor dengan sebuah emulator seperti yang terlihat pada Gambar 1.6. Cukup dengan menginstall program *aiStarter* dan menjalankannya maka emulator akan muncul ketika kita memilih menu *Connect > Emulator*

1.6 Berbagi Aplikasi

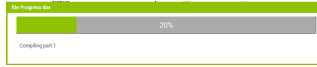
Kita dapat berbagi desain aplikasi yang kita buat dengan melakukan export dan import project seperti yang terlihat pada Gambar 1.7. File akan disimpan dengan ekstensi *.aia* dan selanjutnya dapat dibaca di App Inventor manapun.



Gambar 1.6: Emulator



Gambar 1.7: Berbagi Aplikasi



Gambar 1.8: APK

1.7 Pemaketan Aplikasi

Sebagaimana aplikasi Android yang dikembangkan di Android Studio, App Inventor juga menyiapkan pemaketan aplikasi Android dalam bentuk *APK*. Hal ini dapat dibangun dengan memilih *Build > save.apk to my computer* (Gambar 1.7). Sebelum memilih hal tersebut maka kita harus menjalankan:

Windows jalankan perintah:

```
WinStartBuildServer.cmd
```

Linux jalankan perintah:

```
./startBuildServer.sh
```

Setelah itu, kita perlu memilih menu *Build > save.apk to my computer*, maka akan muncul progress bar seperti pada Gambar 1.8.

BAB 2

YOUR NAME IS ...

Tahukah kamu nama-nama populer di Indonesia? Dari hasil survey[6] terhadap 56.719 orang nama Dwi, Muhammad dan Nur merupakan nama yang sering dipakai oleh orang Indonesia. Mari kita membuat aplikasi App Inventor kita yang pertama yang terkait dengan masalah nama. Aplikasi ini menerima masukan nama kita. Ketika tombol aplikasi ini ditekan maka akan ada suara yang menyebutkan siapa nama kita dan muncul kotak dialog yang menyapa kita.

2.1 Tujuan

- Mengetahui bagaimana membuat aplikasi sederhana dengan App Inventor
- Mengenal *event* pada tombol
- Menampilkan dialog dengan menggunakan *Notifier*
- Memahami penggunaan *TextBox* dan *text-to-speech*

2.2 Pembuatan Program

Kali ini kita akan membuat dua macam aplikasi Android yaitu: menampilkan nama dan menyebutkan nama dari input yang kita berikan.

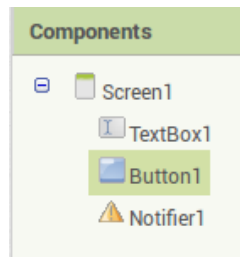
2.2.1 Tampilkan Namamu

Aplikasi Tampilkan Namamu ini hanya memiliki tiga komponen yang dapat dilihat pada Tabel 2.1.

Tabel 2.1: Daftar Komponen Tampilkan Namamu

Komponen	Fungsi
Button	pengguna menekan tombol ini untuk menampilkan namanya
TextBox	menerima input nama
Notifier	menampilkan nama yang diinputkan

Komponen pada Gambar 2.1 harus anda pilih pada App Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut.

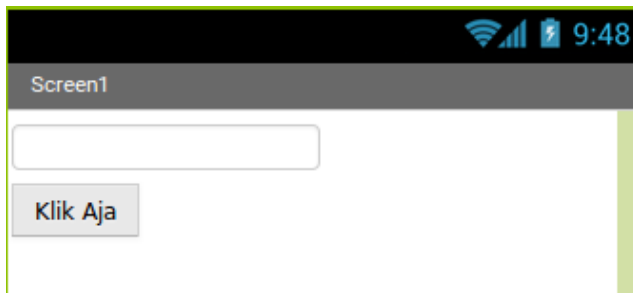


Gambar 2.1: Komponen Tampilkan Namamu pada App Inventor

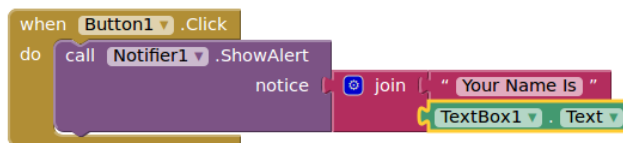
Posisi masing-masing komponen disusun seperti pada Gambar 2.2

Pada bagian *Blocks Editor*, kita perlu memberikan perilaku pada *Button1* saat ditekan oleh pengguna seperti yang terlihat pada Gambar 2.3. Jika *Button1* ditekan, maka *Notifier1* dipanggil untuk menampilkan kalimat *Your name is* dan input dari *TextBox1*.

Hasil dari aplikasi ini dapat dilihat pada Gambar 2.4.



Gambar 2.2: Desain Aplikasi Tampilkan Namamu



Gambar 2.3: Blocks Editor Aplikasi Tampilkan Namamu

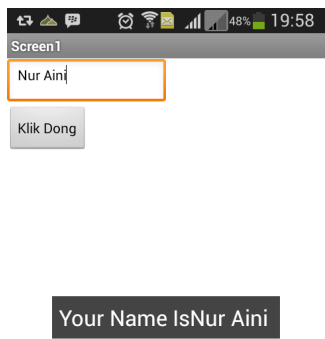
2.2.2 Katakan Namamu

Aplikasi kedua ini hampir sama dengan aplikasi pertama, namun komponen *TextToSpeech* ditambahkan agar smartphone dapat mengucapkan nama yang kita masukkan pada *TextBox1*. Daftar komponen katakan namamu bisa dilihat pada Tabel 2.2.

Detil komponen yang harus diletakkan pada aplikasi tersebut dapat dilihat pada Gambar 2.5.

Hampir sama dengan aplikasi pertama, aplikasi yang satu ini juga menampilkan kalimat dengan menggunakan *Notifier1*, namun aplikasi ini memanggil *TextToSpeech1* untuk mengatakan kalimat *Your name is* dan input dari *TextBox1* seperti yang terlihat pada Gambar 2.6.

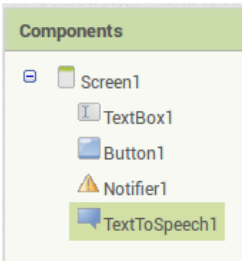
Hasil dari aplikasi ini dapat dilihat pada Gambar 2.4, namun kita dapat mendengar suara yang menyebutkan *Your name is ...*



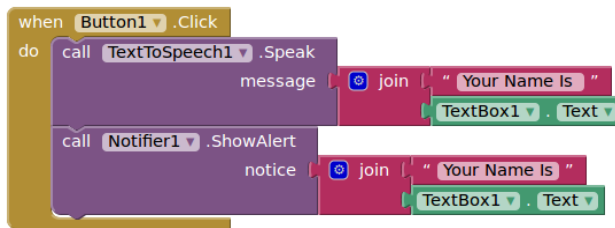
Gambar 2.4: Tampilan Aplikasi Tampilkan Namamu

Tabel 2.2: Daftar Komponen Katakan Namamu

Komponen	Fungsi
Button	pengguna menekan tombol ini untuk menampilkan namanya
TextBox	menerima input nama
Notifier	menampilkan nama yang diinputkan
TextToSpeech	mengucapkan suara sesuai dengan input textbox yang kita berikan



Gambar 2.5: Komponen Katakan Namamu pada App Inventor



Gambar 2.6: Blocks Editor Aplikasi Katakan Namamu

BAB 3

SENYUM DAN KLIK

Senyum membawa dampak yang luar biasa terhadap apa yang kita kerjakan. Sebuah studi yang dilakukan oleh Strack [7] memperlihatkan hubungan senyum dengan gambar yang diciptakan oleh orang yang sedang tersenyum. Sekelompok peserta pada studi ini diminta menggigit sebuah pensil sehingga mereka dipaksa untuk tersenyum, sedangkan sekelompok peserta lainnya diminta meletakkan pensil di mulutnya tanpa mengenai giginya sehingga wajahnya akan terlihat cemberut. Semua peserta diminta untuk menggambar sebuah kartun. Hasil studi tersebut menyebutkan orang yang tersenyum akan menghasilkan kartun lebih lucu daripada orang yang berwajah cemberut. Pada bab ini kita akan membuat aplikasi yang membuat orang tersenyum saat kita mengambil gambarnya.

3.1 Tujuan

1. Mengetahui penggunaan fitur *camera* untuk mengambil gambar
2. Mengetahui penggunaan fitur *video recorder* untuk mengambil gambar
3. Menampilkan video dan gambar pada aplikasi Android
4. Memahami cara perpindahan *screen* pada App Inventor

3.2 Pembuatan Program

Aplikasi ini terdiri atas dua *screen*. Screen pertama berfungsi untuk mengambil gambar dan screen kedua berfungsi merekam

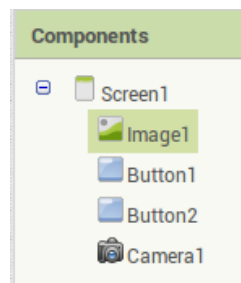
Tabel 3.1: Daftar Komponen Aplikasi Senyum

Komponen	Fungsi
Button	Pengguna menekan tombol ini untuk mengambil gambar, merekam video dan berpindah atas screen
Image	Menampilkan Gambar
Camera	Mengambil Gambar
Screen	Menampilkan seluruh komponen
VideoPlayer	Memutar video
CamCorder	Merekam kejadian dalam bentuk video

video. Kita akan mempelajari perpindahan screen di bab ini. Daftar komponen yang dibutuhkan pada aplikasi Senyum ini terdapat pada Tabel 3.1

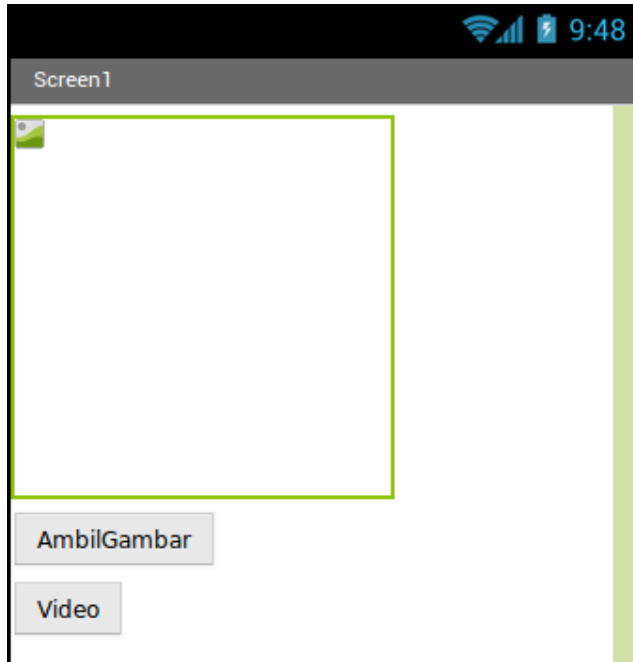
3.2.1 Screen: Ambil Foto

Untuk screen pertama kita akan membutuhkan komponen *Camera* dan *Image*. Untuk lebih lengkapnya pada screen pertama terdapat komponen-komponen yang terdapat pada Gambar3.1. *Button1* berfungsi untuk mengambil gambar dan *Button2* untuk berpindah ke screen kedua.



Gambar 3.1: Desain Komponen Senyum di Screen 1

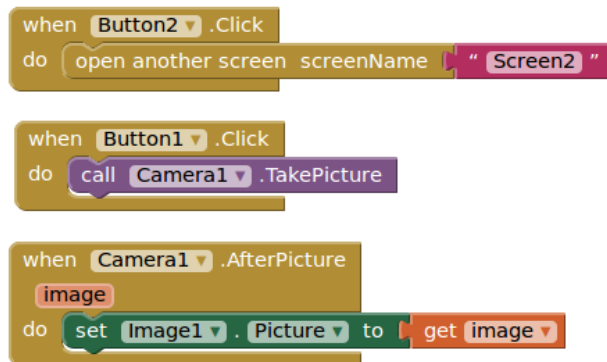
Pada screen pertama, kita mengatur desain aplikasi seperti pada Gambar 3.2. *Button1* diberi label *AmbilGambar* dan *Button2* diberi label *Video*.



Gambar 3.2: Desain Aplikasi Senyum di Screen 1

Pada bagian *Blocks Editor* (Gambar 3.3), kita perlu memberikan perilaku pada *Button1*, *Button2* dan *Camera1*. Jika *Button1* ditekan, maka akan memanggil *Camera1* untuk mengambil gambar. Setelah *Camera1* mengambil gambar, *Image1* akan berisi gambar yang diambil oleh *Camera1*. *Screen2* akan dipanggil saat *Button2* ditekan.

Hasil dari aplikasi ini dapat dilihat pada Gambar 3.4.



Gambar 3.3: Blocks Editor Aplikasi Senyum di Screen 1

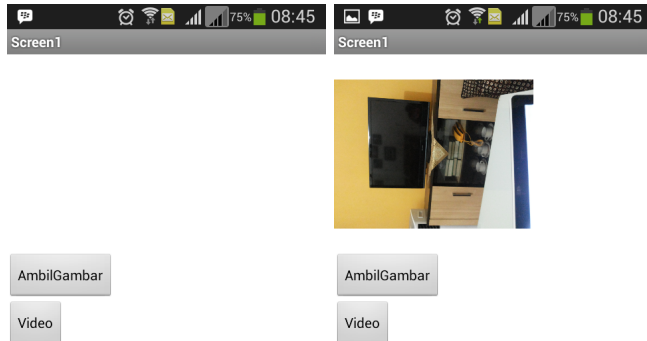
3.2.2 Screen: Rekam Video

Untuk screen kedua kita akan membutuhkan komponen *Video-Player* dan *CamCorder*. Untuk lebih lengkapnya pada screen kedua terdapat komponen-komponen yang terdapat pada Gambar3.5. *Button1* berfungsi untuk merekam kejadian dan *Button2* untuk berpindah ke screen pertama.

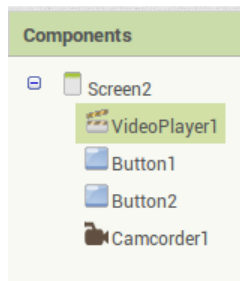
Pada screen kedua, kita mengatur desain aplikasi seperti pada Gambar 3.6. *Button1* diberi label *RekamVideo* dan *Button2* diberi label *Kamera*.

Pada bagian *Blocks Editor* (Gambar 3.7), kita perlu memberikan perilaku pada *Button1*, *Button2* dan *CamCorder1*. Jika *Button1* ditekan, maka akan memanggil *CamCorder1* untuk merekam video. Setelah *CamCorder1* merekam video, *Video-Player1* akan berisi *clip* yang direkam oleh *CamCorder1*. *Screen1* akan dipanggil saat *Button2* ditekan.

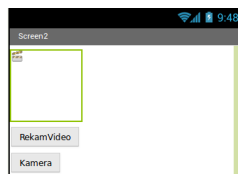
Hasil dari aplikasi ini dapat dilihat pada Gambar 3.8.



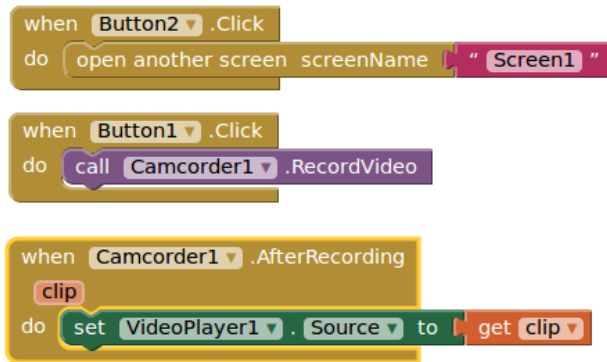
Gambar 3.4: Tampilan Aplikasi Senyum di Screen 1



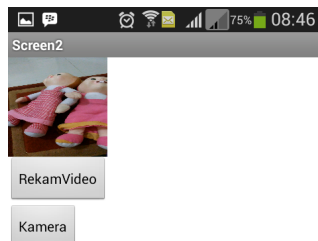
Gambar 3.5: Desain Komponen Senyum di Screen 2



Gambar 3.6: Desain Aplikasi Senyum di Screen 2



Gambar 3.7: Blocks Editor Aplikasi Senyum di Screen 2



Gambar 3.8: Tampilan Aplikasi Senyum di Screen 2

PHANTOM VIBRATIONS SYNDROME

Pernahkah kita merasa HP atau smartphone kita bergetar padahal pada kenyataannya Hp kita tidak bergetar sama sekali? Jangan khawatir ternyata tidak hanya anda yang mengalami halusinasi ini. Sebuah studi [8] menyatakan sebagian besar orang pernah merasa ada panggilan atau notifikasi masuk, namun mereka tidak merasa terganggu dengan keadaan tersebut. Halusinasi ini dikenal dengan nama *Phantom Vibrations Syndrome*. Kali ini kita akan menggetarkan smartphone Android kita jika kita menggerakkan smartphone dengan menggunakan sensor *Accelerometer*.

4.1 Tujuan

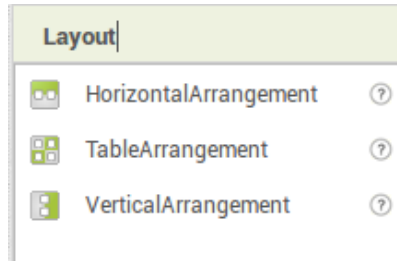
- Mengetahui penggunaan *Accelerometer Sensor* dan *Sound*
- Memahami cara mengatur tampilan di App Inventor
- Mendefinisikan variabel global

4.2 Layout Pada App Inventor

Sebelum membuat aplikasi, mari kita mengenal pengaturan tampilan pada App Inventor. Kita dapat memilih tiga macam layout yaitu *HorizontalArrangement*, *TableArrangement* dan *VerticalArrangement* pada *Palette Layout* (Gambar 4.1).

1. *HorizontalArrangement*

Layout *HorizontalArrangement* menampilkan komponen-komponen dari kiri ke kanan secara horizontal.



Gambar 4.1: Jenis Layout pada App Inventor

2. *TableArrangement*

Layout *TableArrangement* menampilkan komponen-komponen dalam bentuk tabel. Setiap komponen hanya boleh terletak dalam satu sel. Jika banyak komponen terletak dalam satu sel, maka hanya satu komponen yang akan terlihat. Kita perlu mengatur jumlah baris dan kolom pada layout ini.

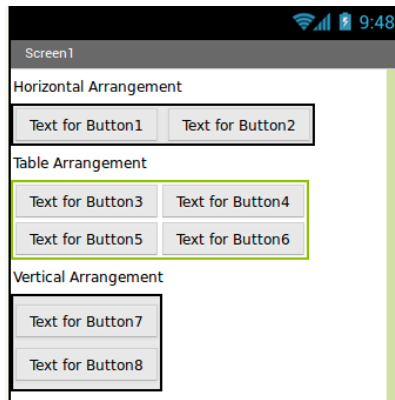
3. *VerticalArrangement*

Layout *VerticalArrangement* menyusun komponen-komponen dari atas ke bawah secara vertikal dan rata kiri.

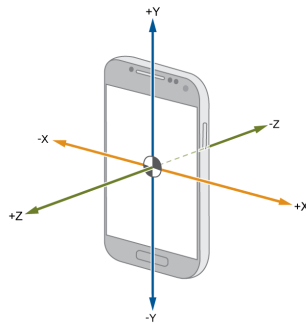
Perbedaan tampilan untuk masing-masing layout dapat dilihat pada Gambar 4.2.

4.3 Accelerometer Sensor

Akselerometer adalah alat untuk mengukur percepatan atau akselerasi sebuah benda. Masih ingatkah tentang percepatan gravitasi ($g = 9.8m/s^2$) saat benda jatuh ke bumi? Di dalam smartphone terdapat juga sensor akselerometer yang dapat memperlihatkan posisi smartphone kita berada dalam bentuk tiga



Gambar 4.2: Tampilan Tiga Jenis Layout



Gambar 4.3: Accelerometer pada smartphone [1]

dimensi. Posisi sumbu X, Y dan Z pada sebuah smartphone dapat dilihat pada Gambar 4.3.

Jika smartphone kita jauh ke tanah maka akan bernilai 0. Sensor ini mempunyai satu event yang dinamakan *AccelerometerSensor.AccelerationChanged* yang mempunyai tiga dimensi berikut:

1. **xAccel**

- Bernilai + jika smartphone digoyang ke kanan.

- Bernilai - jika smartphone digoyang ke kiri.

2. yAccel

- Bernilai + jika bagian bawah smartphone diangkat ke atas
- Bernilai - jika bagian atas smartphone diangkat ke atas

3. zAccel

- Bernilai + jika layar smartphone diletakkan pada posisi atas
- Bernilai - jika layar smartphone diletakkan pada posisi bawah

4.4 Variabel Global

Variabel Global ini akan berubah nilainya pada fungsi-fungsi saat aplikasi dijalankan. Untuk mendeklarasikan sebuah variabel global kita perlu di *Block Editor*. Pilih bagian variabel yang berwarna orange, kemudian pilih *Initialize global **name** to*. Bagian *name* bisa diganti dengan nama variabel yang kita inginkan. Kemudian kita definisikan nilai variabel tersebut dengan angka, string atau boolean.



Gambar 4.4: Deklarasi variabel global

4.5 Pembuatan Program

Sekarang kita mulai membuat dua aplikasi Android yaitu 1) program *GetarZZZ* yang menampilkan perubahan nilai x , y

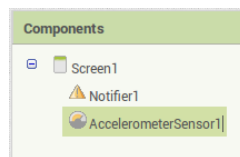
Tabel 4.1: Daftar Komponen Aplikasi Bergoyang

Komponen	Fungsi
Button	Memilih kurun waktu smartphone bergetar
Sound	Berfungsi menggetarkan smart-phone
Notifier	Menampilkan nilai x,y,z yang berubah saat kita gerakan
AccelerometerSensor	Mendeteksi perubahan posisi smart-phone
Layout	Menyusun komponen <i>Button</i>

dan z jika kita menggerakan dan mengoyangkan smartphone kita maka nilainya akan berubah; 2) program *Vibrasi* yang akan membuat smartphone kita bergetar pada kurun waktu yang kita pilih. Kedua program tersebut membutuhkan komponen-komponen yang dapat dilihat pada Tabel 4.1.

GetarXYZ

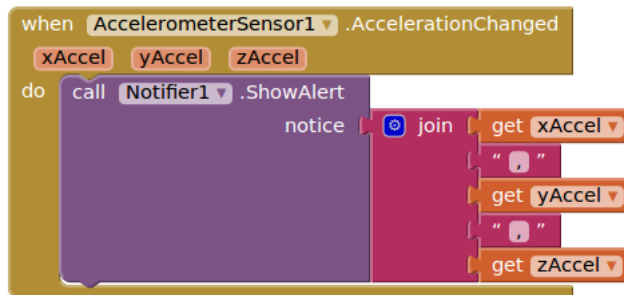
Detil komponen yang harus diletakkan pada aplikasi GetarXYZ dapat dilihat pada Gambar 4.5.



Gambar 4.5: Komponen GetarXYZ pada App Inventor

Pada bagian *Blocks Editor*, kita perlu memberikan perilaku pada *AccelerometerSensor* saat smartphone digoyangkan/digerakkan oleh pengguna seperti yang terlihat pada Gambar 4.6. Jika smartphone digoyangkan/digerakkan, maka *Notifier1* di-

panggil untuk menampilkan kalimat nilai perubahan dari *AccelerometerSensor*. *AccelerometerSensor* menghasilkan tiga luaran (*xAccel*, *yAccel* dan *zAccel*) yang harus digabungkan sebelum dikirim ke *Notifier1*. Oleh karena itu kita perlu melakukan penggabungan tiga luaran tersebut dengan menggunakan fungsi *Join* pada *Text* seperti yang terlihat pada Gambar 4.7. *Join* text bawaannya hanya bisa dua teks, namun kita bisa menambahkan lebih dari dua teks dengan menekan tombol biru seperti di pada Gambar 4.7.



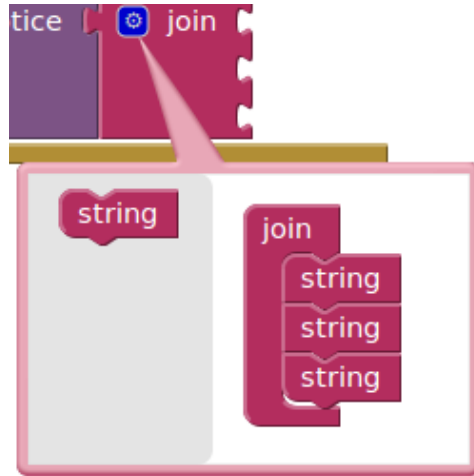
Gambar 4.6: Blocks Editor Aplikasi GetarXYZ

Hasil dari aplikasi ini dapat dilihat pada Gambar 4.8. Gerakkan smartphone anda, maka nilai x,y,z akan berubah.

Vibrasi

Vibrasi adalah sebuah aplikasi yang akan bergetar selama kurun waktu yang kita pilih dari sebuah tombol. Detil komponen yang harus diletakkan pada aplikasi *Vibrasi* dapat dilihat pada Gambar 4.9. Kita memakai layout *TableArrangement* untuk menyusun tombol 100 dan 200.

Posisi masing-masing komponen disusun seperti pada Gambar 4.10.



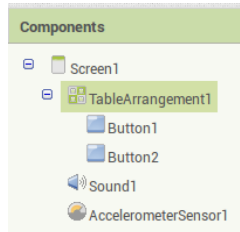
Gambar 4.7: Penggabungan teks

Pada bagian *Blocks Editor*, kita perlu mendefinisikan variabel global *waktu* menjadi 0.5 detik. Kemudian kita beri perilaku pada *Button1* jika ditekan maka variabel global *waktu* menjadi 1 detik. Selain itu, kita perlu memberikan perilaku *Button1* jika ditekan maka variabel global *waktu* menjadi 2 detik. Ketika *AccelerometerSensor.shaking* terjadi maka komponen *Sound1* akan diminta untuk bergetar selama kurun waktu yang dipilih. Rangkuman dari *Blocks Editor* aplikasi Vibrasi ini bisa dilihat pada Gambar 4.11



2.54623,,1.66756,9.56124

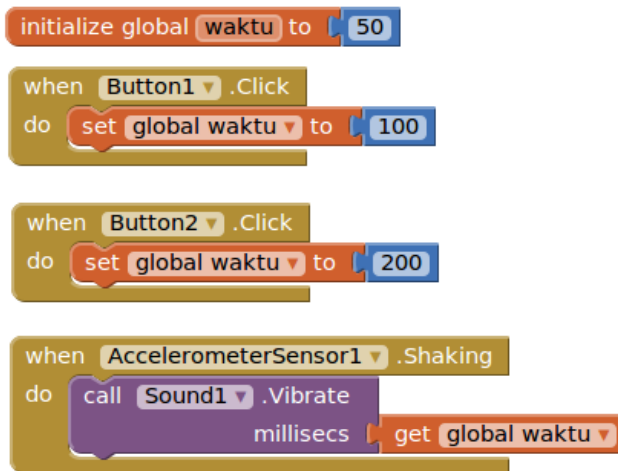
Gambar 4.8: Tampilan Aplikasi GetarXYZ



Gambar 4.9: Komponen Vibrasi pada App Inventor



Gambar 4.10: Desain Aplikasi vibrasi



Gambar 4.11: Blocks Editor Aplikasi Vibrasi

BAB 5

MERAH KUNING HIJAU

Apakah warna melambangkan jenis kelamin kita? Seperti warna pink yang biasanya identik dengan warna wanita. Sebuah studi [9] yang dilakukan oleh Joe Hallock terhadap 232 dari 22 negara memperlihatkan bahwa warna biru menjadi warna favorit pria dan wanita, sedangkan warna ungu menjadi pilihan warna favorit sebagian wanita, namun tidak satupun pria yang menyukai warna ungu ini.

Kali ini kita akan belajar tentang *canvas* pada App Inventor yang meliputi warna dan garis.

5.1 Tujuan

1. Memahami penggunaan Canvas
2. Mempelajari event drag and touch
3. Menggambar garis pada *Canvas*

5.2 Pembuatan Program

Aplikasi Gambar yang kita buat ini mempunyai beberapa komponen yang terdapat pada Tabel 5.1.

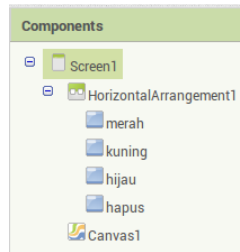
Komponen pada Gambar 5.1 harus anda pilih pada App Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut.

Posisi masing-masing komponen disusun seperti pada Gambar 5.2. *Button* merah, kuning, hijau dan hapus disusun dengan Layout *HorizontalArrangement*.

Pada bagian *Blocks Editor*, kita perlu memberikan perilaku pada *Button* merah, kuning, hijau dan hapus saat ditekan.

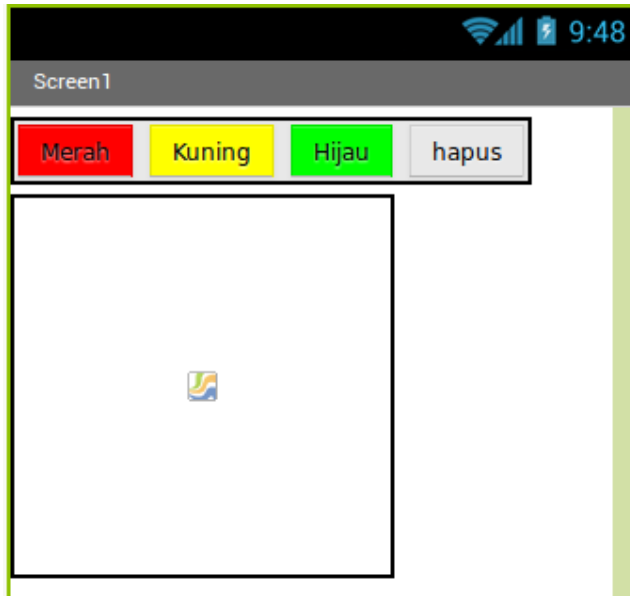
Tabel 5.1: Daftar Komponen Aplikasi Gambar

Komponen	Fungsi
Button	melakukan pemilihan warna
Canvas	bidang dimana kita bisa melakukan penggambaran diatasnya.
Layout	mengatur tombol merah, kuning dan hijau



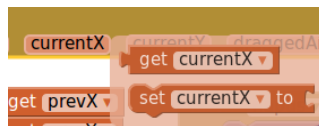
Gambar 5.1: Komponen aplikasi Gambar pada App Inventor

Button merah, kuning, hijau akan mendefinisikan nilai *Canvas1.Paintcolor* sesuai dengan warna *Button* yang ditekan. *Button* hapus akan memanggil fungsi *Canvas1.clear*. Selanjutnya kita perlu memberikan perilaku pada *Canvas1* saat disentuh dan ditarik (*touch and drag*). Untuk mendapatkan posisi sebelum dan sesudah kita menarik tangan dengan bersentuhan pada layar smartphone kita memerlukan empat nilai yaitu *prevX*, *prevY*, *currentX* dan *currentY*. Keempat nilai ini bisa kita dapatkan pada fungsi *When.Canvas.Dragged*. Untuk mendapatkan nilai tersebut, kita tinggal menekan salah satu nilai pada fungsi tersebut hingga keluar seperti pada Gambar 5.3. Penggambaran sebuah garis (*Canvas.DrawLine*) membutuhkan empat nilai tersebut untuk dimasukkan pada variable *x1*, *y1*, *x2* dan *y2*. Ketebalan sebuah garis dapat diseting dengan



Gambar 5.2: Desain Aplikasi Gambar

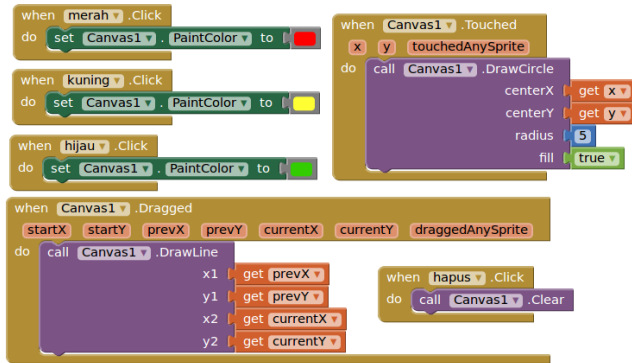
menggunakan *Canvas.Touched* yang memanggil perintah *Canvas.DrawCircle*. Parameter radius pada *Canvas.DrawCircle* ini yang akan mempengaruhi ketebalan dari garis.



Gambar 5.3: Mengambil Parameter

Rangkuman dari *Blocks Editor* aplikasi Vibrasi ini bisa dilihat pada Gambar 5.4

Hasil dari aplikasi ini dapat dilihat pada Gambar 5.5.



Gambar 5.4: Blocks Editor Aplikasi Gambar



Gambar 5.5: Tampilan Aplikasi Warna

BAB 6

DI SINI AKU BERADA !

Saat ini mencari sebuah lokasi sangat mudah dengan keberadaan teknologi *Global Positioning System* (GPS) yang berada di smartphone. Sebuah survey [10] pada tahun 2013 menyatakan bahwa 74% orang dewasa berumur di atas 18 tahun menggunakan smartphonenya untuk memberikan navigasi menuju tempat tertentu. Kali ini kita akan mencoba bermain-main dengan *Location Sensor* yang menggunakan GPS untuk menentukan sebuah lokasi.

6.1 Tujuan

1. Mempelajari penggunaan *Location Sensor*
2. Membagi informasi di smartphone

6.2 GPS

Untuk membuat aplikasi, mari kita belajar tentang GPS. GPS ini mendapatkan data dari satelit. Koordinat posisi kita dibaca oleh GPS yaitu dari lintang, bujur dan ketinggian. Nilai lintang 0 terletak pada kota-kota yang dilewati katulistiwa, sedangkan nilai bujur 0 terletak pada kota Greenwich dekat London, Inggris.

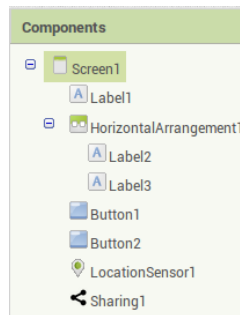
6.3 Pembuatan Program

Aplikasi GPS mirip dengan aplikasi *check-in* dimana GPS membaca posisi kita dan membagikan alamat kita berada ke media lainnya. Aplikasi GPS yang kita buat ini mempunyai beberapa komponen yang terdapat pada Tabel 6.1.

Tabel 6.1: Daftar Komponen Aplikasi GPS

Komponen	Fungsi
Button	Memanggil GPS untuk membaca posisi dan membagikan informasi
Label	Menampilkan posisi lokasi berdasarkan GPS
LocationSensor	membaca posisi
Sharing	membagikan informasi ke sosial media, email dan lain-lain

Komponen pada Gambar 6.1 harus anda pilih pada App Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut.



Gambar 6.1: Komponen aplikasi GPS pada App Inventor

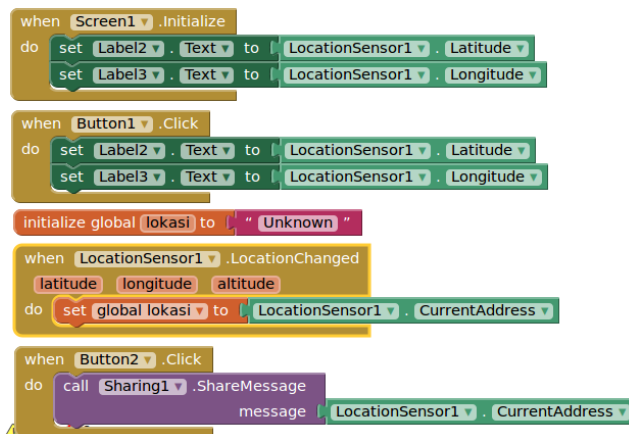
Posisi masing-masing komponen disusun seperti pada Gambar 6.2.

Pada bagian *Blocks Editor*, kita mendeklarasikan variable global yang diberi nama lokasi. Variabel tersebut kita beri nilai "unknown". Variabel ini berubah jika *LocationSensor1* mendeteksi perpindahan posisi (*LocationSensor1.LocationChanged*). Selanjutnya, kita perlu memberikan perilaku pada *Button1* dan *Button2* saat ditekan. *Button1* akan memanggil *LocationSen-*



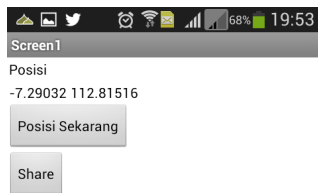
Gambar 6.2: Desain Aplikasi GPS

sensor1.Longitude dan *LocationSensor1.Latitude* untuk ditampilkan pada *Label2* dan *Label3*. *Button2* akan memanggil komponen *Sharing1* untuk berbagi pesan yang berisi alamat lokasi saat ini. Rangkuman dari *Blocks Editor* aplikasi Vibrasi ini bisa dilihat pada Gambar 6.3



Gambar 6.3: Blocks Editor Aplikasi GPS

Hasil dari aplikasi ini dapat dilihat pada Gambar 6.4.
tat



Gambar 6.4: Tampilan Aplikasi GPS

BAB 7

LEMPARKAN BOLA ITU

Olah raga yang menjadi favorit kamu? Menurut [11], 3,5 Bil-yar penduduk dunia terutama di wilayah Eropa, Asia, Afrika dan Amerika paling menyukai olahraga sepak bola. Meskipun dalam piala dunia sepakbola hanya beberapa negara yang berhak mengikuti, namun hampir setiap negara menyiarkan olah raga ini. Di Bab 5 kita telah membahas *Canvas* untuk menggambarkan sesuatu yang statis, kali ini kita akan menampilkan bola yang bergerak di atas sebuah *Canvas*.

7.1 Tujuan

1. Membuat animasi dengan bola
2. Mengenal perilaku bola saat disentuh
3. Mengenal perilaku bola saat menyentuh sudut canvas.

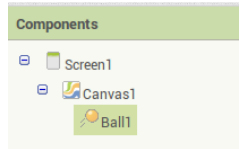
7.2 Pembuatan Program

Aplikasi bola yang kita buat dapat menggerakkan bola dengan sentuhan tangan atau dengan menggoyangkan smartphone kita. Aplikasi bola yang kita buat ini mempunyai beberapa komponen yang terdapat pada Tabel 7.1.

Tabel 7.1: Daftar Komponen Aplikasi Bola

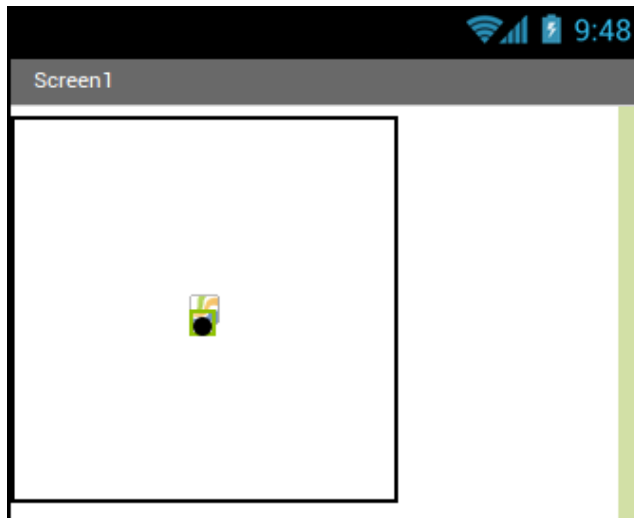
Komponen	Fungsi
Ball	Membuat bola animasi
Canvas	Sebagai bidang untuk bola animasi bergerak

Komponen pada Gambar 7.1 harus anda pilih pada App Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut.



Gambar 7.1: Komponen aplikasi Bola pada App Inventor

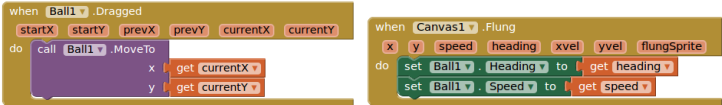
Posisi masing-masing komponen disusun seperti pada Gambar 7.2.



Gambar 7.2: Desain Aplikasi Bola

Pada bagian *Blocks Editor*, kita perlu memberikan perilaku pada saat *Ball* disentuh dan ditarik. Hal tersebut dapat diberikan pada *Ball1.dragged* dengan memanggil perintah *Ball1.MoveTo* ke posisi koordinat saat ini (*currentX* dan *currentY*). Perpindahan bola juga bisa diatur oleh gerakan smart-

phone kita. Untuk keperluan tersebut, kita medefinisikan perilaku bola saat *Canvas1.Flung*. Posisi heading bola dan kecepatan pergeseran bola disesuaikan dengan gerakan smart-phone kita. Rangkuman dari *Blocks Editor* aplikasi bola ini bisa dilihat pada Gambar 7.3



Gambar 7.3: Blocks Editor Aplikasi Bola

Hasil dari aplikasi ini dapat dilihat pada Gambar 7.4.



Gambar 7.4: Tampilan Aplikasi Lokasi

BAB 8

SIMPAN DATAMU

Siapakah yang mempunyai server terbanyak di dunia? Facebook? Google? Atau Microsoft? Menurut CloudTweaks.com [12], Amazon mempunyai jumlah server terbanyak di dunia dengan jumlah sekitar 1,400,000. Kemudian diikuti oleh Google dan Microsoft yang memiliki 1,000,000 server. Server-server tersebut mampu menampung data yang cukup besar. Bagaimana dengan data di smartphone Android? Yuk kita belajar tentang penyimpanan data di Android.

8.1 Tujuan

1. Mempelajari konsep penyimpanan data dengan menggunakan *TinyDB*
2. Mencari data pada TinyDB jika diketahui kata kunci datanya.
3. Mendeklarasikan variabel lokal

8.2 TinyDB

Pada bab-bab sebelumnya kita menggunakan variable global untuk menyimpan data kita. Variabel global ini hanya bersifat aktif saat aplikasi juga sedang aktif. Namun jika aplikasi dimatikan, maka nilainya akan kembali seperti semula. Oleh karena itu, kita perlu menyimpan data kita ke *TinyDB*. *TinyDB* ini database yang sangat sederhana. Data yang tersimpan dalam bentuk string atau teks yang diberi *tag* atau penanda. Contoh:

tag : Keyrani

value : Malang

tag : Syahnaz

value : Surabaya

Tag ini berfungsi sebagai identitas suatu data. Kita bisa memanggil data berdasarkan tag-nya. Satu aplikasi hanya boleh memiliki satu *TinyDB*. *TinyDB* mempunyai dua fungsi utama yaitu *StoreValue* dan *GetValue* (Gambar 8.1).



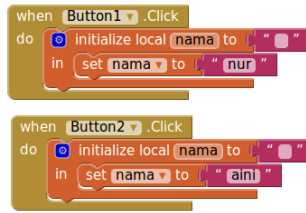
Gambar 8.1: Fungsi TinyDB

8.3 Variabel Lokal

Pada Bab 4 kita telah mempelajari variabel global. Kali ini kita akan memahami variabel lokal. Variabel ini hanya berlaku pada sebuah bagian dari aplikasi. Sebagai contoh Gambar 8.2 terdapat *nama* sebagai variabel lokal pada saat *Button1.Click*. Jika pada *Button2.Click* ingin menggunakan *nama* sebagai variabel, maka kita harus mendeklarasikan variable *nama* terlebih dahulu. Perubahan nilai dari variabel lokal tidak akan mempengaruhi nilai variabel lokal dengan nama yang sama di tempat lain. Misalnya variabel *nama* pada *Button1.Click* dengan variabel *nama* pada *Button2.Click* mempunyai nilai yang berbeda.

8.4 Pembuatan Program

Aplikasi yang akan kita buat adalah aplikasi Kontak yang berfungsi untuk menyimpan nomor HP teman sekaligus melakukan pencarian nomor HP jika diketahui no HP-nya. Aplikasi Kontak



Gambar 8.2: Contoh Variable Lokal

Tabel 8.1: Daftar Komponen Aplikasi Simpan

Komponen	Fungsi
Button	Memanggil <i>TinyDB</i> untuk menyimpan dan mencari kontak
TextBox	Input untuk memasukkan nama dan nomor HP
Label	Menampilkan tulisan Nama dan HP
TinyDB	Menyimpan data kontak

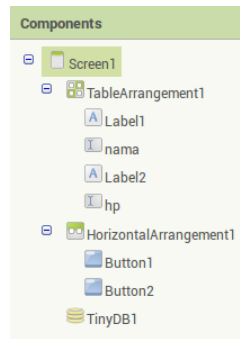
yang kita buat ini mempunyai beberapa komponen yang terdapat pada Tabel 8.1.

Komponen pada Gambar 8.3 harus anda pilih pada App Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut.

Posisi masing-masing komponen disusun seperti pada Gambar 8.4.

Pada bagian *Blocks Editor*, kita mendefinisikan perilaku untuk *Button1* dan *Button2*. Ketika *Button1* disentuh, *TinyDB1.Store Value* dipanggil untuk menyimpan no HP dengan *Nama*. Setelah itu nilai *Textbox* Nama dan HP dibersihkan. Jika *Button2* disentuh maka akan melakukan hal-hal sebagai berikut:

1. Mendeklarasikan *nohp* sebagai variabel lokal
2. *nohp* diberi nilai sesuai dengan nilai yang didapat dari



Gambar 8.3: Komponen aplikasi Kontak pada App Inventor



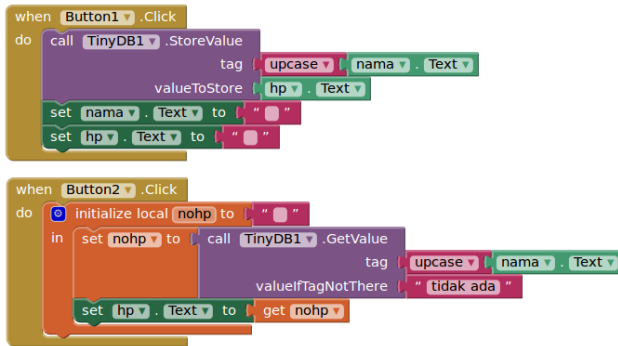
Gambar 8.4: Desain Aplikasi Kontak

TinyDB1 berdasarkan nama yang diberikan

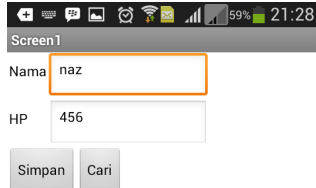
3. Memasukkan nilai dari variabel *nohp* ke *Textbox* HP.

Rangkuman dari *Blocks Editor* aplikasi Kontak ini bisa dilihat pada Gambar 8.5

Hasil dari aplikasi ini dapat dilihat pada Gambar 8.6.



Gambar 8.5: Blocks Editor Aplikasi Kontak



Gambar 8.6: Tampilan Aplikasi Kontak

BAB 9

RUMUS

Rumus matematika adalah sebuah perhitungan yang telah didefinisikan oleh para pakar untuk menghasilkan output dari input yang diberikan. Menghafal rumus matematika saat sekolah terkadang membosankan dan menyulitkan. Mari kita buat rumus matematika di App Inventor dengan input yang diberikan oleh pengguna.

9.1 Tujuan

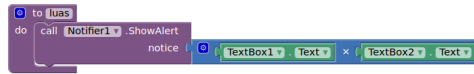
1. Mempelajari *procedure* di App Inventor
2. Membuat *procedure* tanpa parameter dan pengembalian
3. Membuat *procedure* dengan parameter dan tanpa pengembalian
4. Membuat *procedure* dengan parameter dan pengembalian

9.2 Procedure

Procedure adalah bagian dari sebuah program yang kita definisikan sendiri dan dapat dipanggil berulang-ulang untuk memudahkan penulisan program. Ada tiga jenis *procedure* yaitu,

1. *Procedure* tanpa parameter dan pengembalian

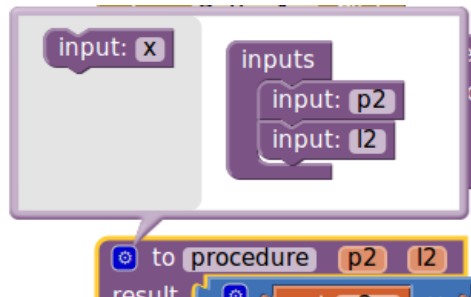
Procedure ini tidak membutuhkan parameter atau input dan tidak mengembalikan output. Contoh pada Gambar 9.1 merupakan *Procedure* perhitungan luas persegi panjang. Luas dan panjang diperoleh dari *Textbox* dan hasilnya langsung ditampilkan dengan *Notifier*.



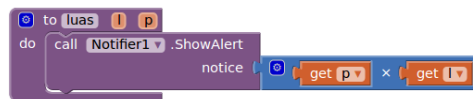
Gambar 9.1: *procedure* tanpa parameter dan pengembalian

2. *Procedure* dengan parameter dan tanpa pengembalian

Procedure luas pada Gambar 9.3 membutuhkan input atau parameter. Untuk membuat *Procedure* yang menerima input, kita perlu mengklik bagian kiri atas *procedure* seperti yang terlihat pada Gambar 9.2. Kita tarik bagian input sesuai dengan jumlah input yang ingin diterima sebuah *Procedure*. *Procedure* luas pada Gambar 9.3 menampilkan output lewat *Notifier*.



Gambar 9.2: Parameter dari sebuah *procedure*



Gambar 9.3: *procedure* dengan parameter dan tanpa pengembalian

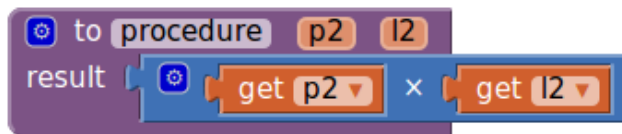
3. *procedure* dengan parameter dan pengembalian

Jenis *Procedure* terakhir ini menerima input dan mengem-

Tabel 9.1: Daftar Komponen Aplikasi Luas

Komponen	Fungsi
Button	memanggil <i>procedure</i> luas
TextBox	menerima input panjang dan lebar
Notifier	menampilkan hasil perhitungan luas
Label	memberi petunjuk isian

balikan output lewat *return*. Contoh pada Gambar 9.4 menerima input panjang dan lebar dan pada bagian *return* mengembalikan isi luas dari persegi panjang.

Gambar 9.4: *procedure* dengan parameter dan pengembalian

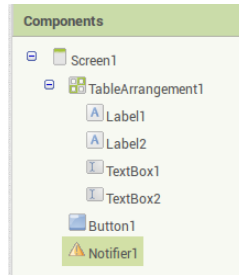
9.3 Pembuatan Program

Mari kita buat aplikasi yang memanggil *procedure* perhitungan luas dari ketiga jenis *procedure* diatas. Aplikasi luas yang kita buat ini mempunyai beberapa komponen yang terdapat pada Tabel 9.1.

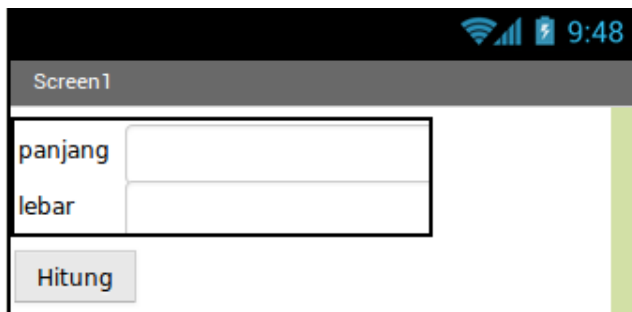
Komponen pada Gambar 9.5 harus anda pilih pada App Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut.

Posisi masing-masing komponen disusun seperti pada Gambar 9.6.

Block Editor pada Gambar 9.7, 9.8 dan 9.9 menghasilkan hasil yang sama yaitu luas sebuah persegi panjang dengan input panjang dan lebar dari *textbox1* dan *textbox2*. Untuk me-



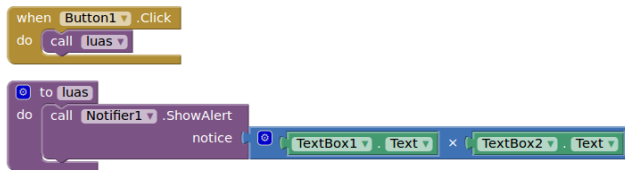
Gambar 9.5: Komponen aplikasi Luas pada App Inventor



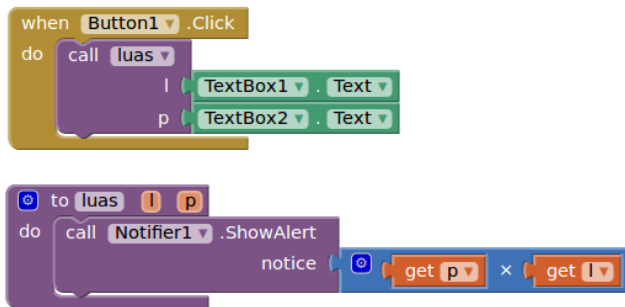
Gambar 9.6: Desain Aplikasi Luas

manggil *procedure* luas pada Gambar 9.7 kita cukup memakai perintah *call luas* saat *Button1.Click*. Sedangkan untuk memanggil *procedure* luas pada Gambar 9.8 dan 9.8 kita harus mendefinisikan nilai panjang dan lebar.

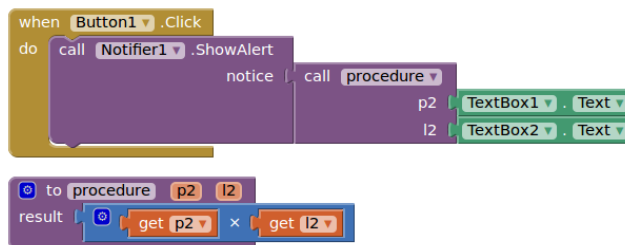
Hasil dari aplikasi ini dapat dilihat pada Gambar 9.10.



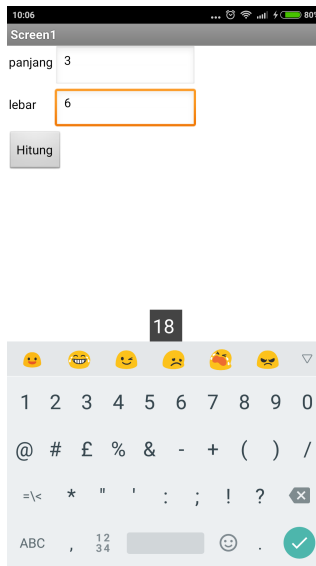
Gambar 9.7: Blocks Editor Aplikasi Luas 1



Gambar 9.8: Blocks Editor Aplikasi Luas 2



Gambar 9.9: Blocks Editor Aplikasi Luas 3



Gambar 9.10: Tampilan Aplikasi Luas

BAB 10

KIRIMKAN PESANMU

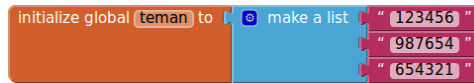
Menurut *Conversational Advertising* [13] bahwa 90% dari SMS yang masuk akan dibaca dalam waktu 3 menit. Tentu saja ini sangat bagus untuk pemasaran suatu produk via sms. Kali ini kita akan belajar menggunakan layanan SMS di smartphone Android.

10.1 Tujuan

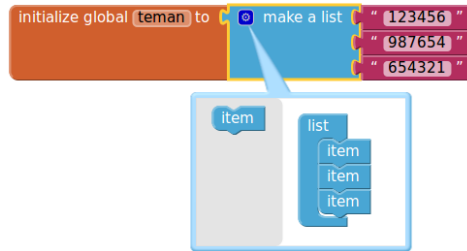
1. Mempelajari pengiriman *SMS* dengan *Texting*
2. Mempelajari konsep *List*
3. Memahami penggunaan *ListPicker*

10.2 List

List adalah sekumpulan data yang disimpan dalam sebuah variabel. Contoh: daftar mahasiswa, daftar kota, dan lain-lain. Sebuah aplikasi dapat mempunyai lebih dari satu *List*. Untuk membuat sebuah *List* baru kita perlu mendeklarasikan variable global *List* dengan menggunakan fungsi *Make a List* seperti pada Gambar 10.1. Jika jumlah data pada *List* lebih dari tiga, maka bisa ditambahkan data baru seperti pada Gambar 10.2. Untuk mengakses data dari sebuah *List*, maka dapat digunakan *select list item list*. Sebagai contoh, untuk mendapatkan nomor teman yang kedua kita bisa menggunakan perintah seperti pada Gambar 10.3.



Gambar 10.1: Contoh Membuat List Baru



Gambar 10.2: Contoh Menambah data List

10.3 Pembuatan Program

Aplikasi SMS yang akan kita buat ini menggunakan *ListPicker* yang berisi data yang berasal dari sebuah *List*. Dengan aplikasi ini kita bisa mengirim SMS dan menambah data teman baru. Aplikasi luas yang kita buat ini mempunyai beberapa komponen yang terdapat pada Tabel 10.1.

Komponen pada Gambar 10.4 harus anda pilih pada App

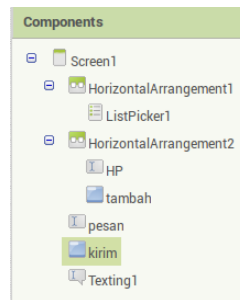
Tabel 10.1: Daftar Komponen Aplikasi SMS

Komponen	Fungsi
Button	Memanggil fungsi mengirim SMS dan menambah data teman baru
TextBox	Menerima input nomor teman dan isi SMS
ListPicker	Menampilkan dan memilih data teman
Texting	Mengirimkan SMS



Gambar 10.3: Contoh Mengakses data di List

Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut.

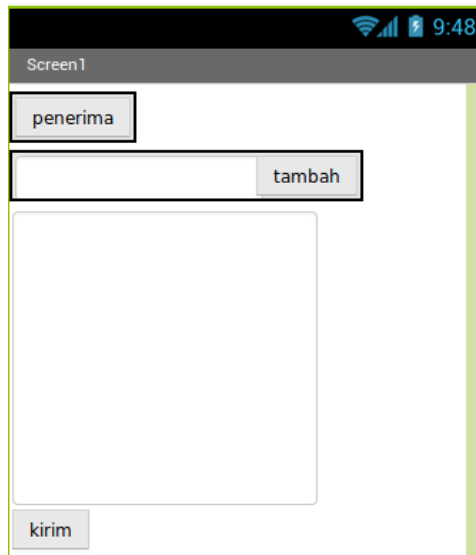


Gambar 10.4: Komponen aplikasi SMS pada App Inventor

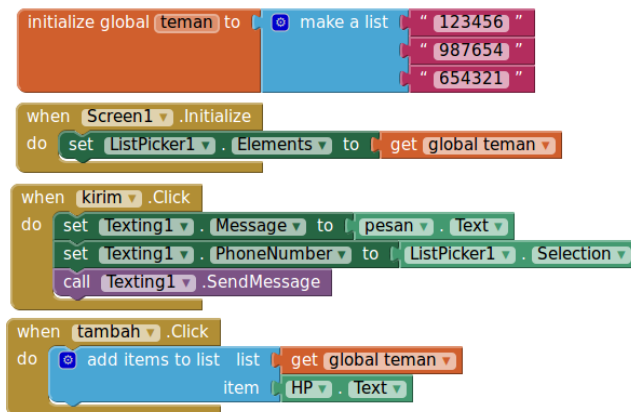
Posisi masing-masing komponen disusun seperti pada Gambar 10.5.

Pada bagian *Blocks Editor*, kita perlu mendeklarasikan *List* teman sebagai variabel global. Setelah *Screen* diinisialisasi, nilai *List* teman masuk ke *ListPicker*. *Button* simpan jika diklik akan memberikan nilai isi pesan dan nomor tujuan ke *Texting1* serta memanggil *Texting1* untuk mengirimkan SMS. Nomor tujuan ini diambil dari *Listpicker*. Nomor tujuan bisa ditambahkan dengan memberikan perilaku pada *Button* tambah, dimana *List* teman akan ditambah dengan data baru dari *Textbox*. Rangkuman dari *Blocks Editor* aplikasi bola ini bisa dilihat pada Gambar 10.6

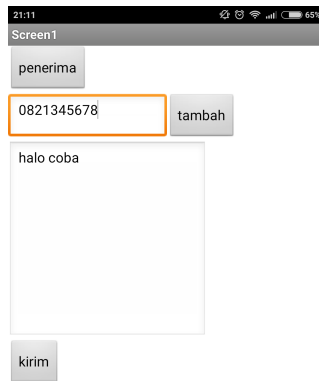
Hasil dari aplikasi ini dapat dilihat pada Gambar 10.7.



Gambar 10.5: Desain Aplikasi SMS



Gambar 10.6: Blocks Editor Aplikasi SMS



Gambar 10.7: Tampilan Aplikasi SMS

MARI BERHITUNG

Banyak perhitungan matematika ternyata menyimpan pola yang cantik. Berikut adalah dua perhitungan matematika yang memiliki pola cantik:

Perkalian barisan 9

$$9 * 9 = 81$$

$$99 * 99 = 9801$$

$$999 * 999 = 998001$$

$$9999 * 9999 = 99980001$$

Perkalian barisan 6 dan 7

$$6 * 7 = 42$$

$$66 * 67 = 4422$$

$$666 * 667 = 444222$$

$$6666 * 6667 = 44442222$$

Pada bab ini kita akan melakukan perhitungan matematika dengan memanfaatkan sebuah *List* sebagai penyimpan seluruh angka yang dimasukkan.

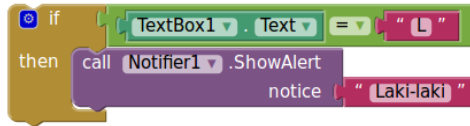
11.1 Tujuan

1. Mempelajari percabangan *if else*
2. Memahami penggunaan *for each* untuk perulangan
3. Mempelajari *List* yang bersifat dinamis

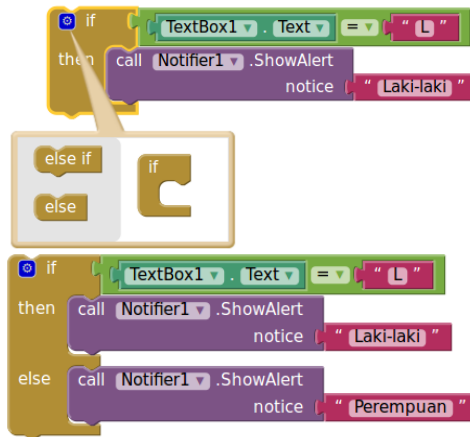
11.2 IF ELSE

IF ELSE berfungsi sebagai percabangan dalam bahasa pemrograman. IF mempunyai sebuah kondisi, dimana jika be-

nar maka perintah selanjutnya akan dikerjakan. ELSE merupakan kondisi jika kondisi IF tidak terpenuhi. Pada Gambar 11.1, jika nilai dari *Textbox1* sama dengan "L" maka *Notifier1* menampilkan "laki-laki". IF pada gambar tersebut tidak mempunyai ELSE sehingga jika tidak terpenuhi, *Notifier1* tidak menampilkan apapun. Untuk itu, kita perlu menambahkan ELSE seperti yang terlihat pada Gambar 11.2. Dengan penambahan ELSE, *Notifier1* akan menampilkan "perempuan" jika kondisinya tidak terpenuhi.



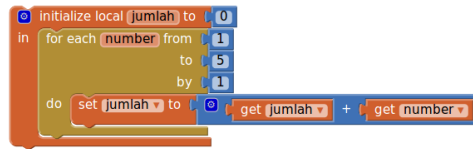
Gambar 11.1: IF



Gambar 11.2: IF ELSE

11.3 FOR EACH

FOR EACH berguna melakukan perulangan saat kondisi masih terpenuhi. Sebagai contoh pada Gambar 11.3 akan dilakukan perulangan dari *number* bernilai 1 hingga 5 dimana nilai *number* bertambah 1 selama perulangan berlangsung. Variabel *jumlah* merupakan variabel lokal yang berfungsi untuk menampung proses penambahan selama perulangan berlangsung.

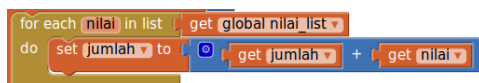


Gambar 11.3: FOR EACH

Untuk lebih jelasnya berikut adalah proses perulangan yang terjadi:

1. $number = 1, jumlah = 0 + 1$
2. $number = 2, jumlah = 0 + 1 + 2$
3. $number = 3, jumlah = 0 + 1 + 2 + 3$
4. $number = 4, jumlah = 0 + 1 + 2 + 3 + 4$
5. $number = 5, jumlah = 0 + 1 + 2 + 3 + 4 + 5$

Selain menggunakan angka seperti pada Gambar 11.3, kita dapat melakukan perulangan untuk mengakses seluruh elemen pada sebuah *List* seperti pada Gambar 11.4. Perulangan dilakukan sebanyak elemen yang terdapat pada *List* nilai *list*, dimana setiap kali perulangan dilakukan, variabel *nilai* akan berisi sebuah elemen dari *nilai_list*.



Gambar 11.4: FOR EACH dan LIST

Tabel 11.1: Daftar Komponen Aplikasi Hitung

Komponen	Fungsi
Button	Memanggil <i>List</i> untuk menambah data dan menghitung rata-rata
TextBox	Menerima input nilai
Notifier	Menampilkan hasil rata-rata nilai yang masuk

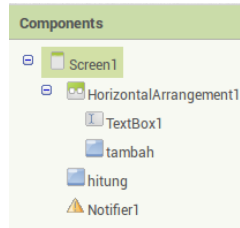
11.4 Pembuatan Program

Aplikasi hitung berfungsi untuk mendapatkan rata-rata nilai dari sekelompok nilai yang dimasukkan. Selanjutnya, aplikasi ini memberikan keputusan apakah kita lulus dengan melihat hasil rata-rata semua nilai yang dimasukkan. Batas nilai rata-rata yang dianggap lulus adalah 50. Aplikasi bola yang kita buat ini mempunyai beberapa komponen yang terdapat pada Tabel 11.1.

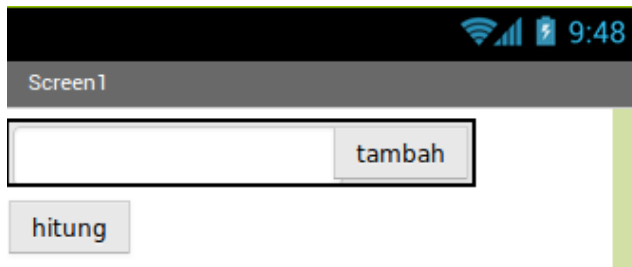
Komponen pada Gambar 11.5 harus anda pilih pada App Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut.

Posisi masing-masing komponen disusun seperti pada Gambar 11.6.

Pada bagian *Blocks Editor*, kita mendefinisikan *nilai_list* sebagai variable global dengan *create empty list*. *nilai_list* akan bertambah jika *Button* tambah disentuh dimana *add list item* dipanggil untuk menambahkan nilai *textbox* ke *nilai_list*. Pada *Button* hitung, kita membutuhkan variabel *jumlah* dan *rata*.



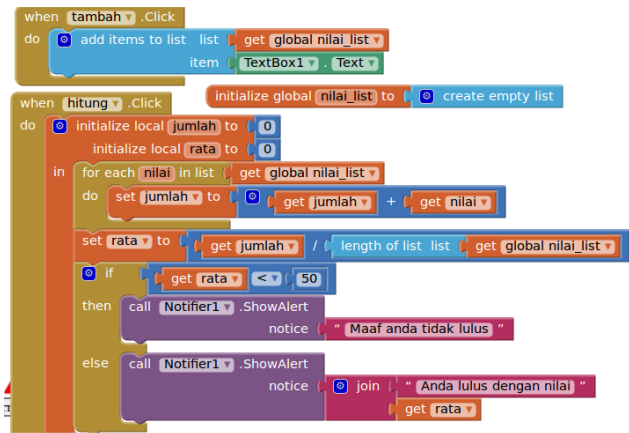
Gambar 11.5: Komponen aplikasi Hitung pada App Inventor



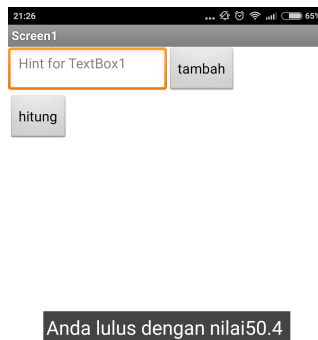
Gambar 11.6: Desain Aplikasi Hitung

Variabel *jumlah* menampung penjumlahan semua elemen pada *nilai_list* selama perulangan di *FOR EACH*. *length list* adalah fungsi untuk mendapatkan berapa jumlah elemen yang ada pada sebuah *LIST*. Setelah nilai rata-rata dihitung, kita perlu mengecek apakah rata-rata di atas 50 dengan menggunakan *IF ELSE*. Rangkuman dari *Blocks Editor* aplikasi bola ini bisa dilihat pada Gambar 11.7

Hasil dari aplikasi ini dapat dilihat pada Gambar 11.8. Coba anda masukkan beberapa nilai dengan menginputkan nilai dan menekan tombol tambah. Setelah itu, tekan tombol hitung dan dapatkan rata-rata semua nilai yang anda masukkan.



Gambar 11.7: Blocks Editor Aplikasi Hitung



Gambar 11.8: Tampilan Aplikasi Hitung

BAB 12

BACA DAN TULIS

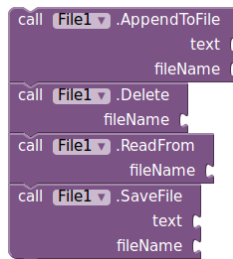
Setelah mempelajari *TinyDB* dan *List* sebagai tempat penyimpanan, kali ini kita akan mempelajari file sebagai salah satu tempat penyimpanan data.

12.1 Tujuan

1. Membaca *file*
2. Menulis *file*
3. Memahami penggunaan *Spinner*

12.2 File

File pada App Inventor akan disimpan di `/AppInventor/data` jika tidak kita definisikan. Ada empat operasi yang bisa kita lakukan pada App Inventor (Gambar 12.1), yaitu membaca (*ReadFrom*), menulis (*Append To File*), menghapus (*Delete*) dan menyimpan (*SaveFile*).



Gambar 12.1: Operasi File pada App Inventor

Tabel 12.1: Daftar Komponen Aplikasi Data Siswa

Komponen	Fungsi
Button	Memanggil perintah untuk menyimpan dan membaca data file
TextBox	Menerima input nama siswa
Spinner	Memilih jenis kelamin
Label	Menampilkan hasil membaca file
File	Tempat penyimpanan

12.3 Pembuatan Program

Kita akan membuat aplikasi data siswa yang terdiri dari nama dan jenis kelaminnya. Jenis kelamin bisa langsung dipilih dengan menggunakan *Spinner*. Data ini disimpan dalam file berbentuk CSV. CSV adalah format penyimpanan dimana setiap kolom dipisahkan dengan tanda koma contoh:

Keyrani,P

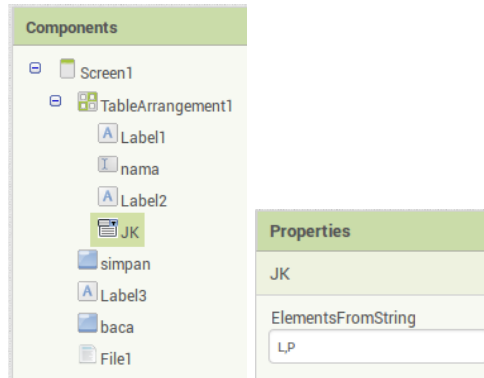
Syahnaz,P

Aplikasi data siswa yang kita buat ini mempunyai beberapa komponen yang terdapat pada Tabel 12.1.

Komponen pada Gambar 12.2 harus anda pilih pada App Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut. Untuk *Spinner* JK kita perlu mendefinisikan *ElementsFromString* dengan nilai *L,P*.

Posisi masing-masing komponen disusun seperti pada Gambar 12.3.

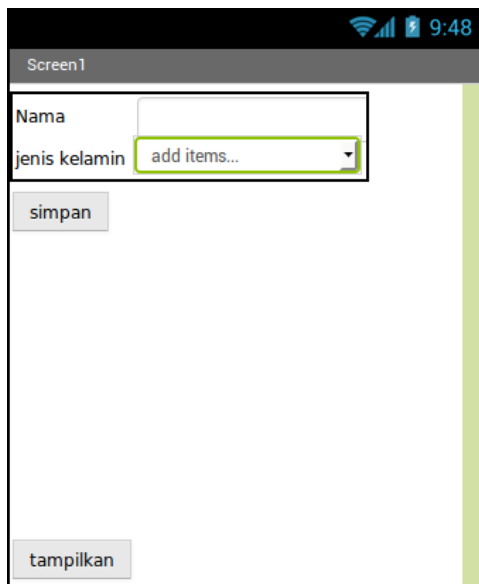
Pada bagian *Blocks Editor*, *Button* simpan akan memanggil perintah *File.AppendToFile*. Nilai dari *Textbox* nama dan *Spinner* JK disimpan ke file yang diberi nama siswa.csv. Untuk mendapatkan nilai *Spinner* kita menggunakan *Spinner.selection*. *Button* baca akan membaca file siswa.csv dengan perintah *Read-*



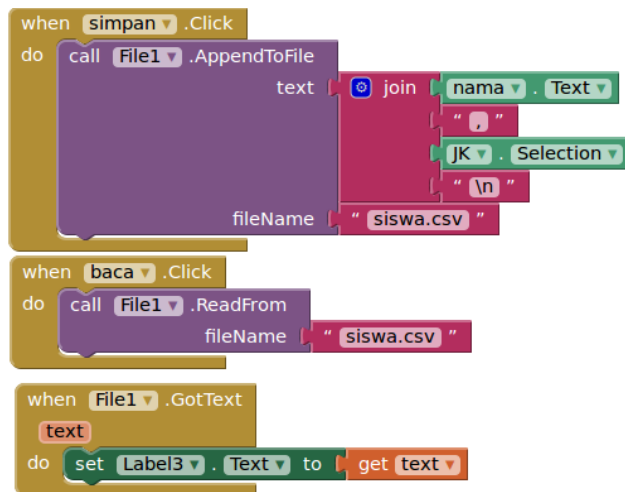
Gambar 12.2: Komponen aplikasi data siswa pada App Inventor

From. Kemudian kita perlu mendeklarasikan operasi jika file siswa.csv ada isinya pada bagian *When File1.GotText*. Jika file ada isinya, maka *Label3* akan menampilkan isi file. Rangkuman dari *Blocks Editor* aplikasi data siswa ini bisa dilihat pada Gambar 12.4

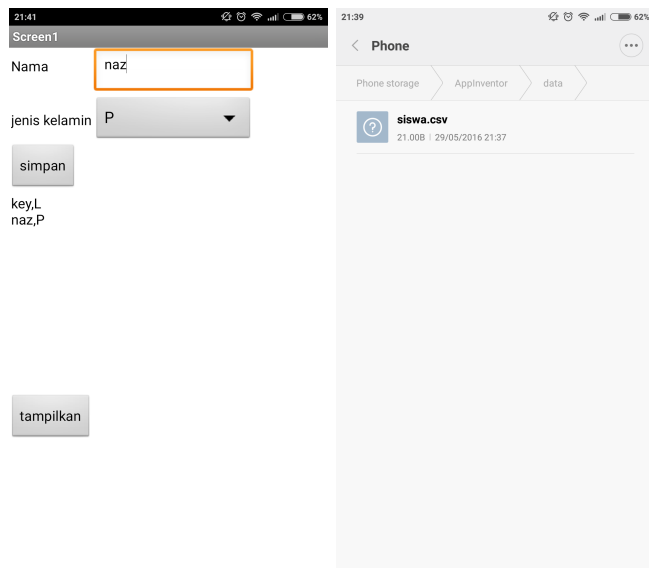
Hasil dari aplikasi ini dapat dilihat pada Gambar 12.5. Cpba kita isikan beberapa nama siswa beserta jenis kelaminnya. Setelah itu, coba tampilkan isinya. Hasil dari data yang kita inputkan dapat dilihat juga pada direktori `/AppInventor/data/-siswa.csv`.



Gambar 12.3: Desain Aplikasi Data Siswa



Gambar 12.4: Blocks Editor Aplikasi Data Siswa



Gambar 12.5: Tampilan Aplikasi Data Siswa

BAB 13

SOTONYA ENAK

Indonesia memang sangat kaya dalam jenis masakan. Sebut saja soto, makanan yang biasanya berkuah kuning ini memiliki jenis yang beragam pada setiap daerahnya seperti soto kudus, soto betawi, soto makasar dan lain-lain. Setiap jenis soto tersebut memiliki resep yang berbeda. Mari kita buat aplikasi untuk menampilkan resep soto dengan *WebView*. Jangan lupa setelah membuat aplikasi ini, pergilah ke dapur dan masak sotonya.

13.1 Tujuan

1. Mempelajari penggunaan *WebView*
2. Menggabungkan *TinyDB* dan *ListPicker*

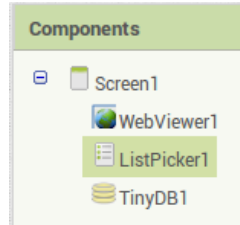
13.2 Pembuatan Program

Aplikasi soto ini menyimpan data alamat resep soto pada *TinyDB*. Alamat akan ditampilkan di *Webviewer* yang berfungsi mirip *browser*. Aplikasi soto yang kita buat ini mempunyai beberapa komponen yang terdapat pada Tabel 13.1.

Tabel 13.1: Daftar Komponen Aplikasi Soto

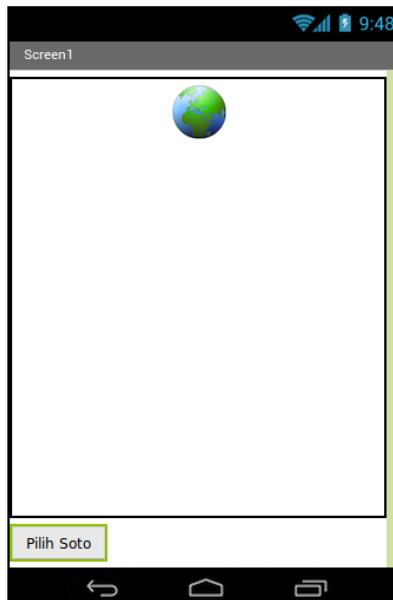
Komponen	Fungsi
TinyDB	Menyimpan data alamat web resep soto dengan tag nama soto
WebView	Menampilkan resep soto
ListPicker	Memilih soto yang ingin ditampilkan di <i>webviewer</i>

Komponen pada Gambar 13.1 harus anda pilih pada App Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut.



Gambar 13.1: Komponen aplikasi Soto pada App Inventor

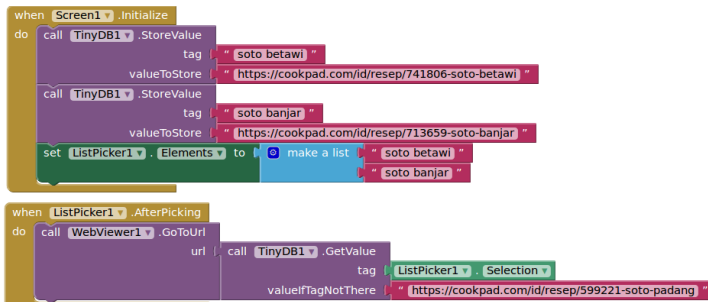
Posisi masing-masing komponen disusun seperti pada Gambar 13.2.



Gambar 13.2: Desain Aplikasi Soto

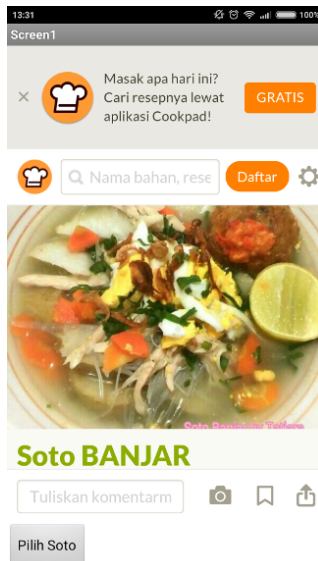
Pada bagian *Blocks Editor*, ketika *Screen* diinisialisasi kita

menyimpan data soto beserta alamatnya di *TinyDB*. Selain itu, kita perlu mendefinisikan isi dari *ListPicker* sesuai dengan nama soto yang ada di *TinyDB*. Kita tidak memerlukan *Button* pada aplikasi soto ini karena *ListPicker* dapat memanggil sebuah operasi ketika salah satu elemennya terpilih dengan menggunakan *ListPicker.AfterPicking*. *ListPicker.AfterPicking* akan memanggil *WebView1* untuk menampilkan alamat resep soto dari *TinyDB* berdasarkan input tag dari *ListPicker*. Rangkuman dari *Blocks Editor* aplikasi bola ini bisa dilihat pada Gambar 13.3



Gambar 13.3: Blocks Editor Aplikasi Soto

Hasil dari aplikasi ini dapat dilihat pada Gambar 13.4. Kita bisa memilih nama soto dan kemudian akan tampil resep soto tersebut.



Gambar 13.4: Tampilan Aplikasi Soto

DENTING WAKTU

Smartphone sudah menjadi bagian yang tak terpisahkan dari hidup kita. Menurut Pew research Center [14], 24% remaja di Amerika Serikat selalu online setiap saat dengan smartphone mereka. Pada risetnya, Lanaj [15] melaporkan bahwa banyak orang menggunakan smartphone menjelang tidur sehingga dapat mengurangi waktu tidurnya. Oleh karena itu, kita harus tetap mengingat waktu dalam menggunakan smartphone. Yuk, kita buat aplikasi waktu sederhana di smartphone Android.

14.1 Tujuan

1. Mempelajari penggunaan *Slider*
2. Mempelajari penggunaan *Clock*
3. Membuat alarm sederhana

14.2 Clock

Ada tiga properti penting pada sebuah *Clock* yaitu:

- *TimerAlwaysFires* akan dijalankan meskipun aplikasi tidak aktif di layar
- *TimerEnabled* akan dijalankan saat aktif saja
- *TimerInterval* untuk menentukan kurun waktu sebuah kejadian dalam mili detik.

Tabel 14.1: Daftar Komponen Aplikasi Waktu

Komponen	Fungsi
Button	Memanggil operasi untuk memulai dan menghentikan waktu
Slider	Mengatur posisi kapan waktu harus berhenti
Label	Menampilkan posisi <i>Slider</i> dan waktu yang sedang berjalan
Clock	Mengitung waktu yang berjalan

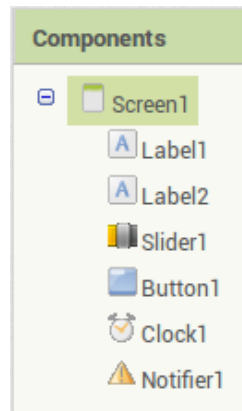
14.3 Pembuatan Program

Aplikasi waktu yang akan kita buat akan menampilkan perpindahan waktu setiap detiknya. Detik akan terus berubah selama belum mencapai posisi yang ditentukan oleh *Slider*. Aplikasi ini mempunyai beberapa komponen yang terdapat pada Tabel 14.1.

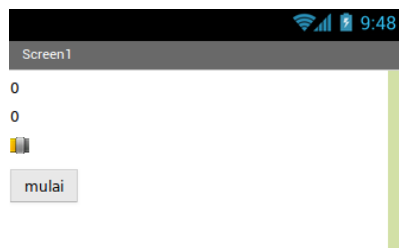
Komponen pada Gambar 14.1 harus anda pilih pada App Inventor. Nama komponen diberi nama sesuai yang terlihat pada gambar tersebut.

Posisi masing-masing komponen disusun seperti pada Gambar 14.2.

Pada bagian *Blocks Editor*, kita perlu mendeklarasi variabel global t dengan nilai 0. t akan bertambah satu persatu saat *Clock* dimulai. Pada saat *Screen* diinisialisasi kita perlu mendefinisikan *TimerAlwaysFires* dan *TimerEnabled* menjadi *false*. *TimerAlwaysFires* dan *TimerEnabled* pada *Clock1* baru akan dijadikan *True* saat *Button1.click*. Kita perlu mendefinisikan *When Slider1.PositionChanged* dimana posisi *Slider* akan ditampilkan pada *Label1* dan nilai dari *Label2* menjadi 0. Ketika *Clock.Timer* dijalankan setiap detiknya, maka operasi berikut



Gambar 14.1: Komponen aplikasi Waktu pada App Inventor



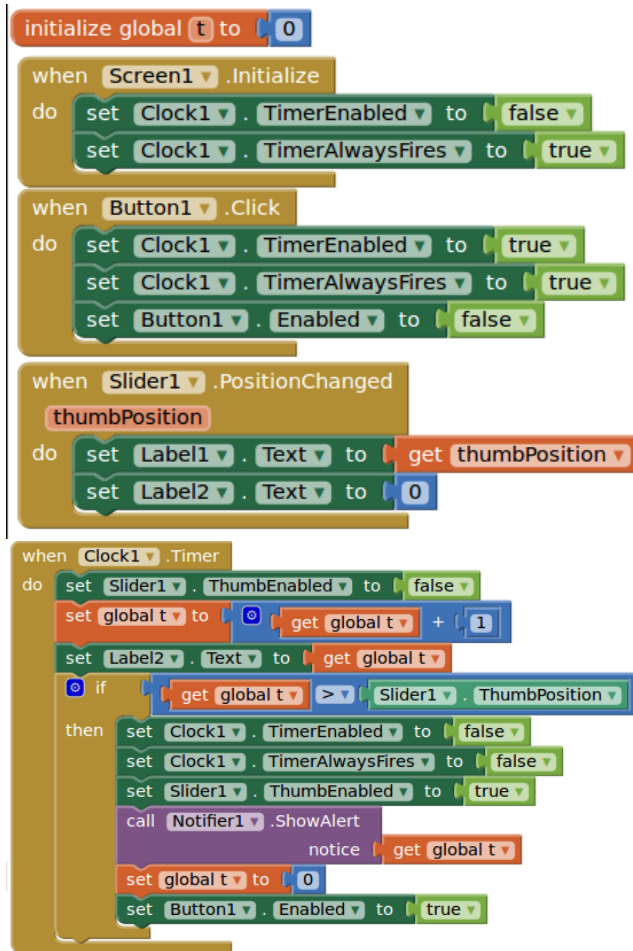
Gambar 14.2: Desain Aplikasi Waktu

dilakukan:

- mendisable posisi *Slider*
- menambahkan nilai variable global t
- memeriksa apakah nilai t sudah melebihi nilai posisi dari *Slider*. Jika nilai t telah melebihi posisi *Slider*, maka:
 - *TimerAlwaysFires* dan *TimerEnabled* pada *Clock* menjadi *false*
 - *Slider* dan *Button* diaktifkan menjadi *True*

- *Notifier* menampilkan nilai t
- nilai $t = 0$

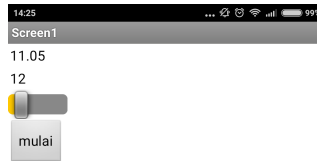
Rangkuman dari *Blocks Editor* aplikasi waktu ini bisa dilihat pada Gambar 14.3



Gambar 14.3: Blocks Editor Aplikasi Waktu

Hasil dari aplikasi ini dapat dilihat pada Gambar 14.4. Geser-

lah slidernya untuk menentukan kapan detik waktu terhenti. Tekan tombol mulai dan perhatikan detik waktu yang tampil dan terhenti saat melebihi waktu di *Slider*.



12

Gambar 14.4: Tampilan Aplikasi Waktu

DAFTAR PUSTAKA

- [1] “Measure linear acceleration along X, Y, and Z axes - Simulink.” Dapat diakses di <http://www.mathworks.com/help/supportpkg/android/ref/accelerometer.html>.
- [2] “Android - History.” Dapat diakses di <https://www.android.com/history/>.
- [3] StatCounter, “Top 8 Mobile & Tablet Operating Systems in Indonesia from Apr 2015 to Mar 2016 | StatCounter Global Stats.” Available online at <http://gs.statcounter.com/#mobile+tablet-os-ID-monthly-201504-201603-bar>.
- [4] “Android Game Apps Are Becoming As Profitable As Those On The iPhone - Business Insider.” Dapat diakses di <http://www.businessinsider.co.id/android-app-profitability-v-ios-2015-1/?r=US&IR=T#ixzz3QEd59GRN>.
- [5] S. C. Pokress and J. J. D. Veiga, “Mit app inventor: Enabling personal mobile computing,” *arXiv preprint arXiv:1310.2830*, 2013.
- [6] “16 Nama Ini Populer di Indonesia, Namamu Termasuk Nggak? - Bangka Pos.” Dapat diakses di <http://bangka.tribunnews.com/2015/12/09/16-nama-ini-populer-di-indonesia-namamu-termasuk-nggak>.
- [7] F. Strack, L. L. Martin, and S. Stepper, “Inhibiting and facilitating conditions of the human smile: a nonobtrusive

- test of the facial feedback hypothesis,” *Journal of Personality and Social Psychology*, vol. 54, pp. 768–777, May 1988.
- [8] R. Rosenberger, “An experiential account of phantom vibration syndrome,” *Computers in Human Behavior*, vol. 52, pp. 124–131, Nov. 2015.
- [9] “True Colors Infographic - Breakdown of Color Preferences by Gender.” Dapat diakses di <https://blog.kissmetrics.com/gender-and-color/>.
- [10] K. Zickuhr, “Location-Based Services,” Sept. 2013. Dapat diakses di <http://www.pewinternet.org/2013/09/12/location-based-services/>.
- [11] S. Desk, “top 10 most popular sports in the world (list),” Sept. 2014. Dapat diakses di <http://www.sportyghost.com/top-10-most-popular-sports-in-the-world/>.
- [12] “Surprising Facts and Stats About The Big Data Industry.” Dapat diakses di <http://cloudtweaks.com/2015/03/surprising-facts-and-stats-about-the-big-data-industry/>.
- [13] “23 Crazy Mobile Marketing Stats - Mobile Is A Must,” Oct. 2014. Dapat diakses di <http://mobilemarketingengine.com/23-mobile-marketing-stats/>.
- [14] A. Lenhart, “Teens, Social Media & Technology Overview 2015,” *Pew Research Center: Internet, Science & Tech*, Apr. 2015.

- [15] K. Lanaj, R. E. Johnson, and C. M. Barnes, “Beginning the workday yet already depleted? consequences of late-night smartphone use and sleep,” *Organizational Behavior and Human Decision Processes*, vol. 124, no. 1, pp. 11–23, 2014.

INDEKS

A

Accelerometer 23
Accelerometer Sensor ...22
AI Companion 6
Android Studio 2
animasi 39
APK 10

B

Blocks Editor 4
Button 12

C

CamCorder 17
Camera 17
Canvas 31
Clock 73
Component Designer 3
CSV 65

E

emulator 8

F

file 64
for each 58

G

GPS 35

H

HorizontalArrangement 22

I

if else 58

J

Java 2
JDK 5

L

List 53, 58
ListPicker 53, 69
Location Sensor 35

N

Notifier 12

P

Palette 22
procedure 47

Q

QR Code 6

S

screen 16
Sharing 36
Slider 73
SMS 53
Sound 22

Spinner 64

T

TableArrangement 23

TextBox 12

Texting 53

TextToSpeech 13

TinyDB 42, 69

V

variabel global 22

variabel lokal 43

VerticalArrangement ... 23

VideoPlayer 17

W

WebView 69