

Abram Hindle
Department of Computing Science
University of Alberta

MVC and Android

MVC Framework

Slides originally by Ken Wong

Images reproduced in these slides have been included under section 29 of the Copyright Act, as fair dealing for research, private study, criticism, or review. Further distribution or uses may infringe copyright.

Who is in Control?

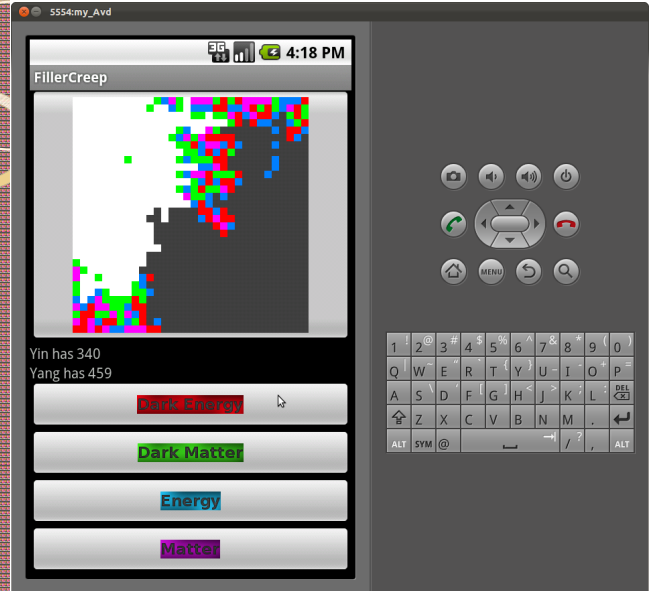
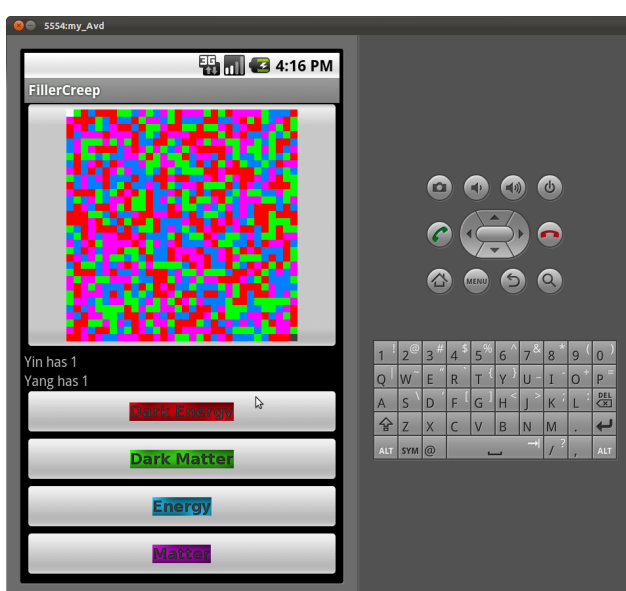
- Class library reuse
 - application developers:
 - write the main body of the application
 - reuse library code by calling it
- Framework reuse
 - application developers:
 - reuse the main body of the application
 - write code that the framework calls
 - reuse library code by calling it

Framework

- Separation of concerns:
 - framework
 - skeletal application code
 - general superclasses and interfaces
 - your “customizations”
 - specific subclasses and implementations

Exercise

- Design an MVC framework for building interactive applications.



Filler Creep Game

- The universe is filled with stuff
- You (Yin) fight Yang for the fundamental stuff that forms the universe.
- You can only consume what you touch
- You will beat Yang if you consume more than Yang.
- 4 kinds of stuff: energy, matter, dark matter and dark energy (I guess you're some of space)
- <https://github.com/abramhindle/FillerCreepForAndroid>

Filler Creep Game

- We're going to use MVC
- Model
 - The universe and game rules
- Views
 - Text View, Graphical View
- Controller
 - Game interaction rules
 - Access to model
 -

The Generic Model

```
public class FModel<V extends FView> {
    private ArrayList<V> views;
    public FModel() {
        views = new ArrayList<V>();
    }
    public void addView(V view) {
        if (! views.contains(view)) {
            views.add(view);
        }
    }
    public void deleteView(V view) {
        views.remove( view );
    }
    public void notifyViews() {
        for (V view : views) {
            view.update( this );
        }
    }
}
```

The Generic View

```
public interface FView<M> {
    public void update( M model);
}
```

The Less Than Generic Controller

```
// The purpose is to decouple the Views
// from the Model and save them from
// changes made to the model
public interface FController {
    public boolean isGameOver();
    public int[] getScores();
    public int whichPlayerNumberWins();
    public Player[] getPlayers();
    public Bitmap getMapBitmap();
    public void playRound(FundamentalStuff
choice);
    public String [] getGameScoreStrings();
}
}
```

The Application

- The application in Android allows us to save local state in memory without communicating through intents.
- We have our singletons here. We will forget them if the application terminates.
- Need to add the application class name in the android.xml

```
<application
    android:name="FillerCreepApplication"
    android:icon="@drawable/ic_launcher"
    android:label="@string/app_name" >
```

The Application

```
public class FillerCreepApplication extends Application {
    // Singleton
    transient private static FillerCreep fillerCreep = null;

    static FillerCreep getFillerCreep() {
        if (fillerCreep == null) {
            fillerCreep = new FillerCreep();
        }
        return fillerCreep;
    }
    // Singleton
    transient private static GameController gameController = null;

    public static GameController getGameController() {
        if (gameController == null) {
            gameController = new GameController(getFillerCreep());
        }
        return gameController;
    }

    @Override
    public void onCreate() {
        super.onCreate();
    }
}
```

Our Model

```
FillerCreep
  nPlayers
  stuffArray
  fillFlood(FundamentalStuff[], int, int, Fundar
  getStuffArray(): FundamentalStuff[]
  inBounds(FundamentalStuff[], int, int): bool
  stackfulFillFlood(FundamentalStuff[], int, in
  stacklessFillFlood(FundamentalStuff[], int, in
  height
  players
  scores
  universe
  width
  FillerCreep()
  FillerCreep(int, int)
  cloneUniverse(): FundamentalStuff[]
  fillFlood(int, int, FundamentalStuff, Fundament
  gameOver(): boolean
  getHeight(): int
  getPlayers(): Player[]
  getScores(): int[]
  getUniverse(): FundamentalStuff[]
```

```
FillerCreep(int, int)
cloneUniverse(): FundamentalStuff[]
fillFlood(int, int, FundamentalStuff, Fundament
gameOver(): boolean
getHeight(): int
getPlayers(): Player[]
getScores(): int[]
getUniverse(): FundamentalStuff[]
getWidth(): int
inBounds(int, int): boolean
init(): void
playAIPlayer(int): int
playPlayer(int, FundamentalStuff): int
playPlayer(Player, FundamentalStuff): int
playRoundWithAI(int, FundamentalStuff): int
resetGame(): void
testPlayerPlay(int, FundamentalStuff): int
testPlayerPlay(Player, FundamentalStuff): int
updateScore(Player, int): void
whichPlayerNumberWins(): int
whichPlayerWins(): Player
```

An example View

```
public class FillerCreepGraphicalViewActivity extends Activity implements
    FView<FillerCreep> {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.graphicalview);
        ImageButton button = (ImageButton) findViewById(R.id.maingraphicalview);
        OnClickListener listener = new OnClickListener() {
            public void onClick(View v) {
                finish();
            }
        };
        button.setOnClickListener(listener);

        FillerCreep fc = FillerCreepApplication.getFillerCreep();
        fc.addView(this);

        updateMap();
    }
    @Override
    public void onDestroy() {
        super.onDestroy();
        FillerCreep fc = FillerCreepApplication.getFillerCreep();
        fc.deleteView(this);
    }
    public void update(FillerCreep fillerCreep) {
        updateMap();
    }
    public void updateMap() {
        ImageButton button = (ImageButton) findViewById(R.id.maingraphicalview);
        GameController gc = FillerCreepApplication.getGameController();
        Bitmap bitmap = gc.getMapBitmap();
        button.setImageBitmap(bitmap);
    }
}
```

An example View

```
public class FillerCreepGraphicalViewActivity extends Activity implements FView<FillerCreep> {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.graphicalview);
        ImageButton button = (ImageButton) findViewById(R.id.maingraphicalview);
        OnClickListener listener = new OnClickListener() {
            public void onClick(View v) {
                finish();
            }
        };
        button.setOnClickListener(listener);

        FillerCreep fc = FillerCreepApplication.getFillerCreep();
        fc.addView(this);

        updateMap();
    }
    @Override
    public void onDestroy() {
        super.onDestroy();
        FillerCreep fc = FillerCreepApplication.getFillerCreep();
        fc.deleteView(this);
    }
    public void update(FillerCreep fillerCreep) {
        updateMap();
    }
    public void updateMap() {
        ImageButton button = (ImageButton) findViewById(R.id.maingraphicalview);
        GameController gc = FillerCreepApplication.getGameController();
        Bitmap bitmap = gc.getMapBitmap();
        button.setImageBitmap(bitmap);
    }
}
```

Example View/Controller

```
public class FillerCreepTextViewActivity extends Activity implements
    FView<FillerCreep> {
    /** Called when the activity is first created. */
    @Override
    public void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.textinterface);

        Button button = (Button) findViewById(R.id.textdarkenergy);
        button.setOnClickListener(new OnClickListener() {
            @Override
            public void onClick(View arg0) {
                play(new DarkEnergy());
            }
        });
        ...
        updateScores();
        FillerCreep fc = FillerCreepApplication.getFillerCreep();
        fc.addView(this);
    }
    public void update(FillerCreep fillerCreep) {
        updateScores();
    }
    @Override
    public void onDestroy() {
        super.onDestroy();
        FillerCreep fc = FillerCreepApplication.getFillerCreep();
        fc.deleteView(this);
    }
    void play(FundamentalStuff choice) {
        GameController gc = FillerCreepApplication.getGameController();
        gc.playRound(choice);
    }
    void updateScores() {
        TextView score1 = (TextView) findViewById(R.id.textyin);
        TextView score2 = (TextView) findViewById(R.id.textyang);
        TextView[] tscores = new TextView[] { score1, score2 };
        GameController gc = FillerCreepApplication.getGameController();
        String[] scores = gc.getGameScoresStrings();
        for (int i = 0; i < tscores.length; i++) {
            tscores[i].setText(scores[i]);
        }
    }
}
```

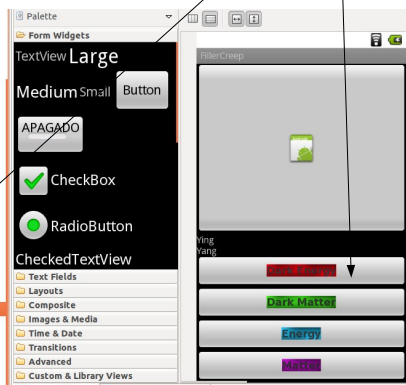
Each Activity Must Be Declared!

```
From AndroidManifest.xml
<activity
    android:name=".FillerCreepActivity"
    android:label="@string/app_name" >
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
<activity>
<activity
    android:name=".FillerCreepTextViewActivity"
    android:label="@string/app_name" >
    <intent-filter>
        </intent-filter>
    </activity>
<activity
    android:name=".FillerCreepGraphicalViewActivity"
    android:label="@string/app_name" >
    <intent-filter>
        </intent-filter>
    </activity>
<activity
    android:name=".FillerCreepGraphicalGameActivity"
    android:label="@string/app_name" >
    <intent-filter>
        </intent-filter>
    </activity>
</activity>
```

ImageButton!

- FillerCreepActivity.java
- FillerCreepApplications.java
- FillerCreepGraphicalGameActivity.java
- FillerCreepGraphicalViewActivity.java
- FillerCreepTextViewActivity.java
- FModel.java
- FundamentalStuff.java
- FundamentalStuffColorMap.java
- FView.java
- GameController.java
- GraphicalFillerCreepView.java
- IntPoint.java
- Matter.java
- Player.java
- Yang.java
- Yin.java
- gen [Generated Java Files]
- Android 2.1
- bin
- res
 - drawable-hdpi
 - dark_energy.png
 - dark_matter.png
 - energy.png
 - ic_launcher.png
 - matter.png
 - drawable-ldpi



Exercise

- Design an MVC framework for building interactive applications.