GWT Architectural Patterns

Dr. Lofi Dewanto
https://lofidewanto.blogspot.de

Agenda

- Model View Controller (MVC) and Model View Presenter (MVP)
- Activities, Places and History Management
- Task
- Modularization and Turducken
- Task

- *Deferred Binding and Generators*
- *Reactive Pattern*

MVC, MVP
Activities, Places and History Management

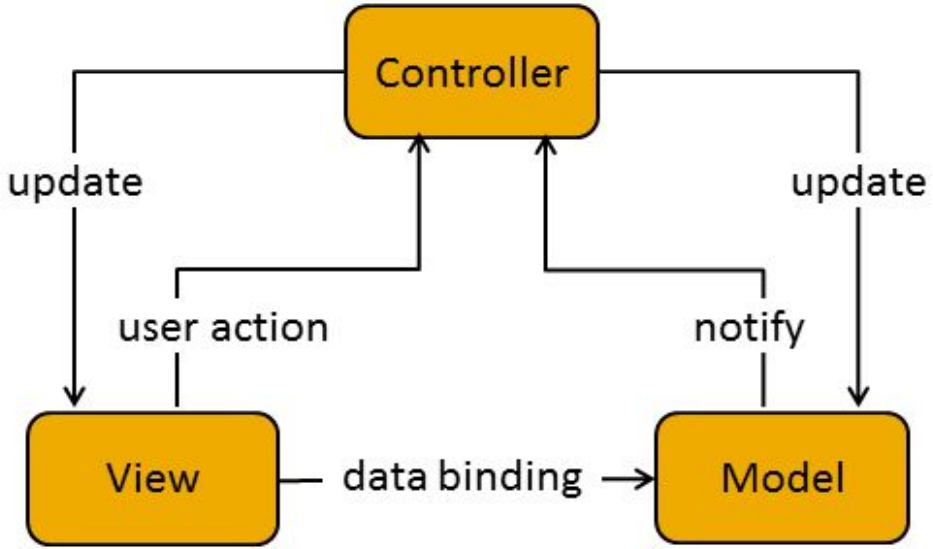
MVC and MVP - Goals

Testability
Application logic can be tested with plain JVM test cases

Maintainability
Simple changes easy, complex changes possible

Source: https://www.youtube.com/watch?v=kilmaSRq49g

Model View Controller (MVC)



Source: https://help.sap.com/erp_hcm_ias2_2014_03/helpdata/en/91/f233476f4d1014b6dd926db0e91070/content.htm?no_cache=true

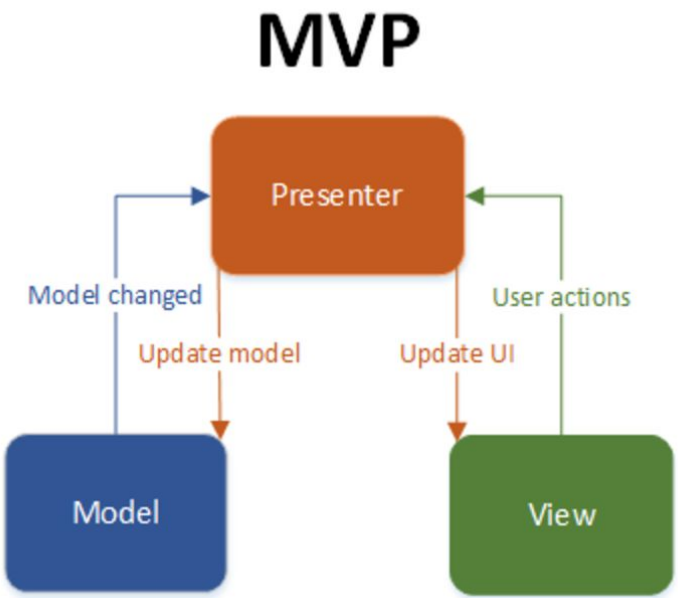
Model View Controller (MVC)

- The ***model*** is responsible for managing the data of the application. It receives user input from the controller.
- The ***view*** means presentation of the model in a particular format.
- The ***controller*** responds to the user input and performs interactions on the data model objects. The controller receives the input, optionally validates it and then passes the input to the model

- Examples: Apache Struts, Spring MVC

Source: https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller

Model View Presenter (MVP)



Source: https://www.techyourchance.com

Model View Presenter (MVP)

- MVP is an **evolved** version of MVC.
- The ***model*** is an interface defining the data to be displayed or otherwise acted upon in the user interface.
- The ***view*** is a passive interface that displays data (the model) and routes user commands (events) to the presenter to act upon that data.
- The ***presenter*** acts upon the model and the view. It retrieves data from repositories (the model), and formats it for display in the view, doing UI logic.

- Examples: GWT, Android, Windows Form, iOS

Source: https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93presenter

Model View Presenter (MVP)

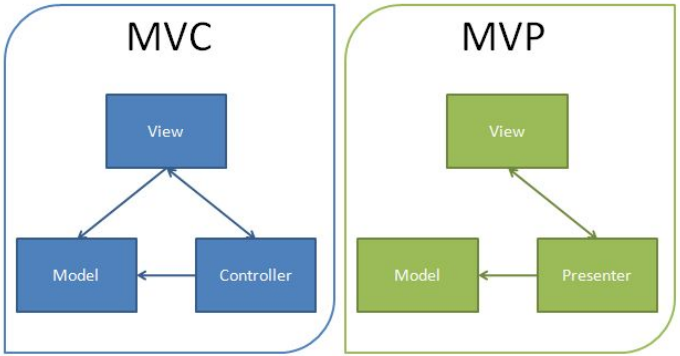
- GWT simple example: https://github.com/lofidewanto/gwt-helloworld-mvp

Model View Presenter (MVP)

- GWT simplify MVP: http://docs.huihoo.com/google/io/2013/421-demystifying-mvp-and-eventbus-in-gwt.pdf

- GWT simplify MVP: https://www.javacodegeeks.com/2012/02/gwt-mvp-made-simple.html

MVP vs. MVC



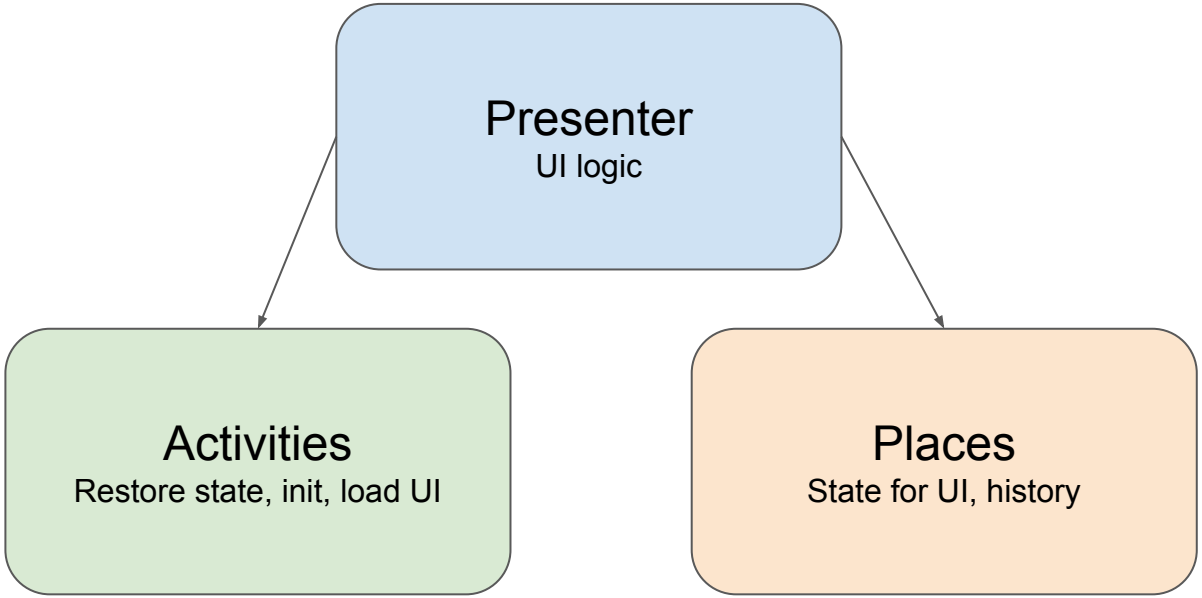
MVP	MVC
Advanced form of MVC	One of the old patterns used to achieve separation of concerns.
View handles the user gestures and calls presenter as appropriate	Controller handles the user gestures and commands model.
View is completely passive, All interactions with model must pass through presenter	View has some intelligence. It can query the model directly
Highly supports unit testing	Limited support to unit testing

Source: https://www.codeproject.com/Articles/288928/Differences-between-MVC-and-MVP-for-Beginners
Source: http://www.g-widgets.com/2016/01/19/introduction-to-the-mvp-pattern-in-gwt

Activities, Places and History Management

- Activities:
  - An activity simply represents something the user is doing. An Activity contains no Widgets or UI code. Activities typically restore state (“wake up”), perform initialization (“set up”), and load a corresponding UI (“show up”). Activities are started and stopped by an ActivityManager associated with a container Widget. An Activity can automatically display a warning confirmation when the Activity is about to be stopped (such as when the user navigates to a new Place). In addition, the ActivityManager warns the user before the window is about to be closed.
- Places:
  - A place is a Java object representing a particular state of the UI. A Place can be converted to and from a URL history token (see GWT’s History mechanism) by defining a PlaceTokenizer for each Place, and GWT’s PlaceHistoryHandler automatically updates the browser URL corresponding to each Place in your app.

Activities, Places and History Management



Activities, Places and History Management

- Example Android Activity

KOTLIN JAVA

```java
public class MainActivity extends AppCompatActivity {
    public static final String EXTRA_MESSAGE = "com.example.myfirstapp.MESSAGE";
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }

    /** Called when the user taps the Send button */
    public void sendMessage(View view) {
        Intent intent = new Intent(this, DisplayMessageActivity.class);
        EditText editText = (EditText) findViewById(R.id.editText);
        String message = editText.getText().toString();
        intent.putExtra(EXTRA_MESSAGE, message);
        startActivity(intent);
    }
}
```

Activities, Places and History Management

- Example GWT: https://github.com/lofidewanto/mvpexample
- Description in detail: http://www.gwtproject.org/doc/latest/DevGuideMvpActivitiesAndPlaces.html

- MVP framework for GWT:
  - https://mvp4g.github.io/mvp4g
  - https://github.com/arcbees/gwtp

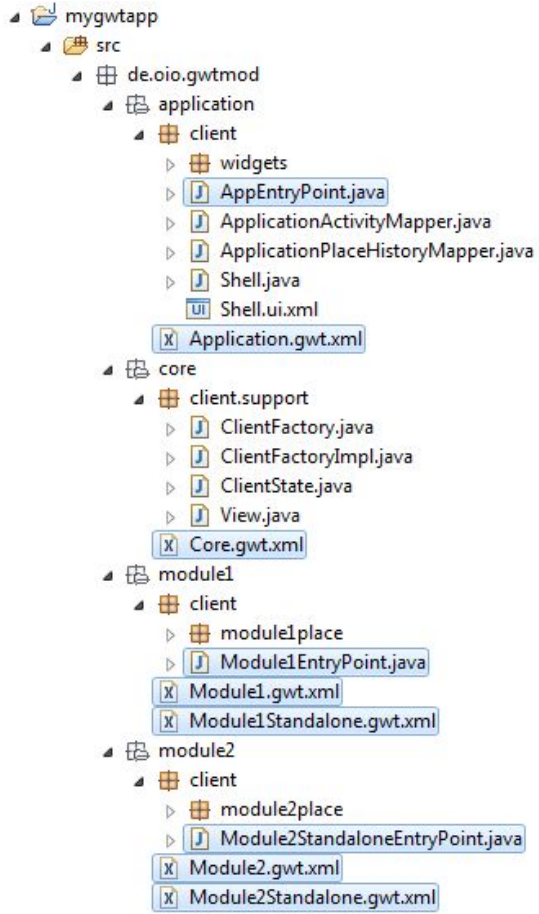
Task

- We have following example: https://github.com/lofidewanto/mvpexample
- Please add from the MainPage:
  - A new View: LogoutView with UIBinder
  - A new Activity: LogoutActivity
  - A new Place: LogoutPlace
  - From LogoutPlace you can go to LoginPlace

Modularization
and
Turducken

Modularization and Turducken

- Module in GWT
- Also possible to separate into many Maven modules
- Remember:
  - GWT is a transpiler and needs the source codes.
  - Everytime it will compile the whole source codes.
- Only separating GWT modules **does not help** to increase the compile / transpile velocity.



Source: https://blog.oio.de/2013/10/14/how-to-split-up-gwt-applications-into-modules

Turducken

- Complex GWT apps can involve **multiple teams** with different release cycles. Compile times can quickly become prohibitive when your codebase grows into millions of lines.

- **“Turducken”** is a design pattern to combine multiple GWT apps that can be built and released by separate teams while providing a seamless, snappy user experience

- **Tur**key + **Duc**k + Chic**ken**

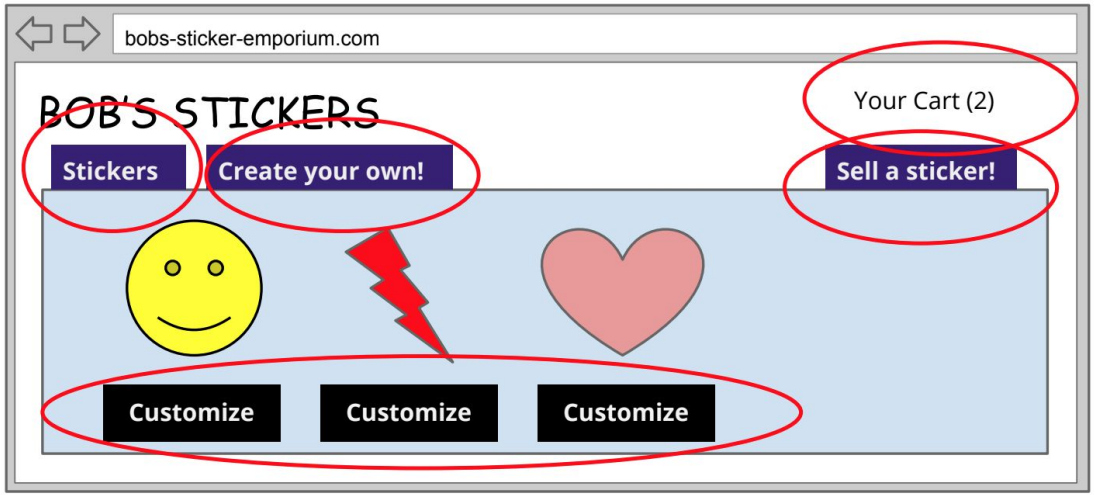
Source: https://de.slideshare.net/RobertKeane1/turducken-divide-and-conquer-large-gwt-apps-with-multiple-teams

Turducken

GWT.create

Multiple modules

Split into multiple GWT entry points



Turducken

- Presentation GWT Create: https://de.slideshare.net/RobertKeane1/turducken-divide-and-conquer-large-gwt-apps-with-multiple-teams

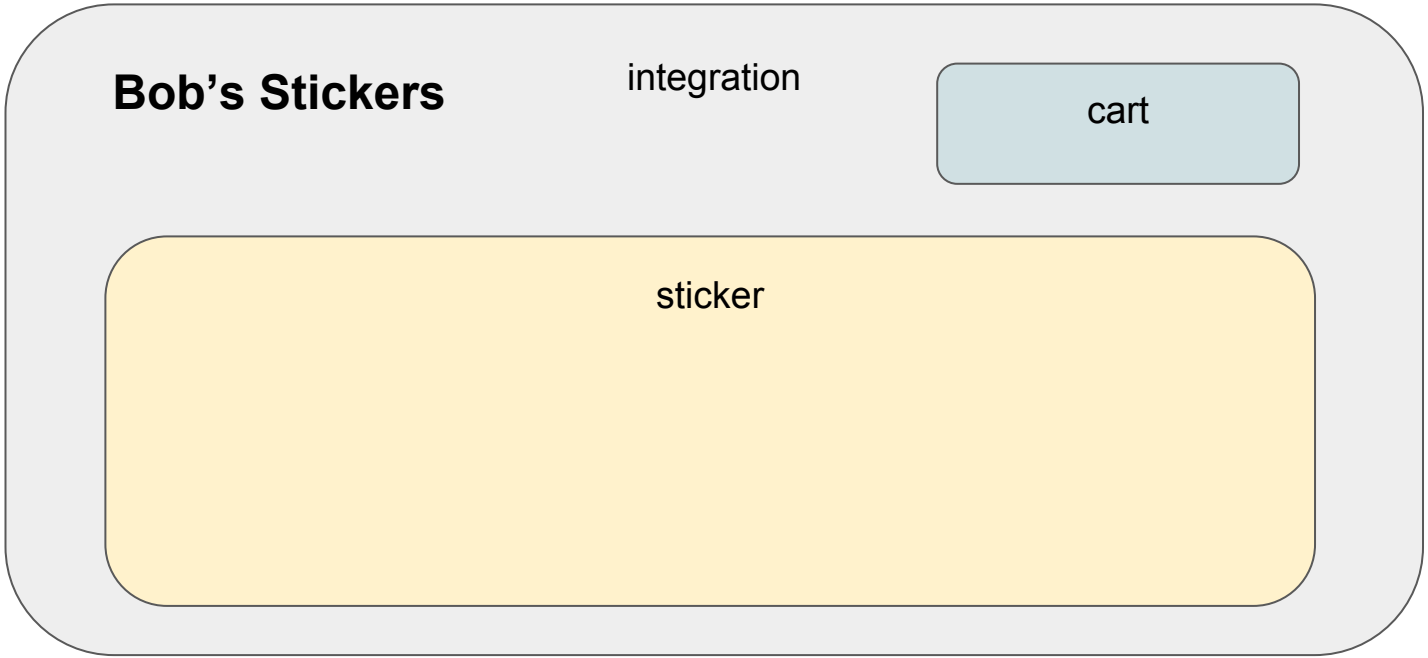
- Presentation GWT Conf: https://de.slideshare.net/gwtcon/diy-split-gwt-applications-using-turducken-approach-by-alberto-mancini

- InterAppEventBus:
  - https://github.com/FrankHossfeld/InterAppEventBus
  - https://github.com/sambathl/interapp-eventbus

Task

- Build three Maven projects / modules with each an Entry Point:
  - Module artifactId - module GWT: **sticker**
  - Module artifactId - module GWT: **cart**
  - Module artifactId - module GWT: **integration**
- In Module integration create a HTML index file which includes two other modules sticker and cart:
  - The integration should be done by adding those two JavaScripts “sticker” and “cart” into the “integration” HTML index file.

Task



Generators and Deferred Binding

Deferred Binding

- Deferred binding is a feature of the GWT compiler that works by generating **many versions of code at compile time**, only one of which needs to be loaded by a particular client during bootstrapping at runtime.
- For example, if you were to internationalize your application using GWT's Internationalization module, the GWT compiler would generate various versions of your application per browser environment, such as "*Firefox in English*", *"Firefox in French"*, *"Internet Explorer in English"*, etc…
- As a result, the deployed JavaScript code is **compact** and **quicker to download than hand coded JavaScript,** containing only the code and resources it needs for a particular browser environment.

Deferred Binding

- Type of Deferred Binding
  - **Replacement:** A type is replaced with another depending on a set of configurable rules.
  - **Code generation:** A type is substituted by the result of invoking a code generator at compile time.

- Documentation: http://www.gwtproject.org/doc/latest/DevGuideCodingBasicsDeferred.html

Generators

- GWT offers a Generator Framework.
- The second technique for deferred binding consists of using generators.
- Generators are classes that are **invoked by the GWT compiler to generate a Java implementation of a class during compilation**. When compiling for production mode, this generated implementation is directly translated to one of the versions of your application in JavaScript code that a client will download based on its browser environment.

Generators

- GWT Generators are **deprecated** for GWT 3
- Use **Java Annotation Processor** instead!
  - Creating / Generating Java codes based on annotations

- Article about Java Annotation Processor: https://medium.com/@iammert/annotation-processing-dont-repeat-yourself-generate-your-code-8425e60c6657

Reactive Pattern

Future and CompletableFuture (Promise)

- Problem: Async Callback
  - Not predictable
  - No order
  - Ugly cascading calls
  - “Callback hell”
- Solution: wrap async callback with **Promise**

Reactive Pattern

- General concept: Reactive pattern and Reactive programming
  - Intro: https://dzone.com/articles/5-things-to-know-about-reactive-programming
  - http://reactive.how
  - https://dzone.com/articles/rxjava-part-1-a-quick-introduction

References

- http://docs.huihoo.com/google/io/2013/421-demystifying-mvp-and-eventbus-in-gwt.pdf
- https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller
- https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93presenter
- https://msdn.microsoft.com/de-de/library/cc749751.aspx
- https://medium.com/@cervonefrancesco/model-view-presenter-android-guidelines-94970b430ddf
- https://android.jlelse.eu/android-mvp-for-beginners-25889c500443
- http://www.g-widgets.com/2016/01/19/introduction-to-the-mvp-pattern-in-gwt
- https://www.codeproject.com/Articles/288928/Differences-between-MVC-and-MVP-for-Beginners
- https://stackoverflow.com/questions/2056/what-are-mvp-and-mvc-and-what-is-the-difference
- https://www.javacodegeeks.com/2012/02/gwt-mvp-made-simple.html
- https://help.sap.com/erp_hcm_ias2_2014_03/helpdata/en/91/f233476f4d1014b6dd926db0e91070/content.htm?no_cache=true
- https://medium.com/ios-os-x-development/ios-architecture-patterns-ecba4c38de52
- http://www.g-widgets.com/2016/02/05/browser-history-management-in-gwt-using-activity-and-place
- https://entwickler.de/online/google-web-toolkit-ui-entwicklung-mit-dem-activities-and-places-framework-114961.html
- https://ronanquillevere.github.io/2013/03/03/activities-places-intro.html#.WzERC_ll9hE

References

- https://de.slideshare.net/RobertKeane1/turducken-divide-and-conquer-large-gwt-apps-with-multiple-teams
- https://de.slideshare.net/gwtcon/diy-split-gwt-applications-using-turducken-approach-by-alberto-mancini
- https://blog.arcbees.com/2015/05/26/how-to-write-gwt-generators-efficiently
- http://www.classicode.net/2017/02/gwts-asynccallback-and-promises
- http://www.gwtproject.org/doc/latest/DevGuideCodingBasicsDeferred.html