GWT Architectures and Lessons Learned

Dipl. Wi.-Ing. Papick Garcia Taboada

pgt technology scouting GmbH
http://pgt.de
Orientation in Objects GmbH
http://oio.de
GWT Development
- Basics
- Structuring the UI
- Talking to the server
Architecture shift

Web applications

Model 2 web applications

Rich internet applications

+ testability
+ maintainance
+ product development
Shift happened

Java development,
JS deployment,
Async,
RPC,
RIA/ single page,
...
what is a JS application?

a chunk of JS that does a lot of DOM manipulation to create web applications
Web 2.0 development is all about...

...js
...html
...css
none of it

...jcp
...oracle
...IBM
...backend
what is a GWT application?

a chunk of JS that does a lot of DOM manipulation to create web applications that was originally written in Java
GWT short 10s introduction
Widget $t = \text{new Label}(\text{"Hello world"});$

RootPanel.get().add($t$);
Architecture shift
Browser

Server

user action

full html response

user action

full html response

user action

full html response

classic web development
what does that mean?

• RIA?
• Architecture shift?
you need fewer resources?

- LEMNISCUS is running on aws T2 micro instances
- 1024mb RAM
- 20% of one CPU + bursts
- bursts account depends on usage, fills up if app under 20% usage
- in fact running stateless on 2 instances for fail over/ redundancy
you can handle more sessions?

• 100 concurrent sessions, so far so good
• we still can scale out
<table>
<thead>
<tr>
<th>Metric</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Average Latency</td>
<td>58.0</td>
</tr>
<tr>
<td>Sum Requests</td>
<td>29.7K</td>
</tr>
<tr>
<td>CPU Utilization</td>
<td>1.4%</td>
</tr>
<tr>
<td>Max Network In</td>
<td>17MB</td>
</tr>
<tr>
<td>Max Network Out</td>
<td>4MB</td>
</tr>
</tbody>
</table>
Web frameworks
low level, generic tools
let's build big things
TextBox t0 = new TextBox();
TextBox t1 = new TextBox();
TextBox t2 = new TextBox();
TextBox t3 = new TextBox();
TextBox t4 = new TextBox();
VerticalPanel...
SplitPanel...
ScrollPanel
RootPanel.get().add(mainPanel);
maintainance hell
MacGyver

All he needed was a ballpoint pen and a paper clip
Ops
Widget t = new Label("hello world");

RootPanel.get().add(t);

not enough...
app-framework wizardry needed
GWT DEVELOPMENT IS COMPLICATED
HOT NEW STUFF.
NOT REALLY
GWT development is not new, but different

WHY?
WEB DEVELOPMENT IS NOT NEW...
2) RICH CLIENT DEVELOPMENT IS NOT NEW EITHER
nothing new here
really NOTHING new here
Look! YES! This is my code!
It’s all about software engineering
Just a few tips
USE MVP!

You will get used to it
USE MVP!

You will get used to it
USE MVP!

You will get used to it

MODEL

PRESENTER

MOCK VIEW

unit test this
USE MVP!

You will get used to it
event bus please

http://jarrettws.blogspot.de/2010/05/public-transport.html
eventbus.fireEvent(
  NotificationEvent.info(
    "Daten wurden erfolgreich gespeichert"
  )
);

![Notification displaying "Daten wurden erfolgreich gespeichert"]
SINGLETON

Don't public static instance
BUT ON IE 6 IT IS SO SLOW!
BUT ON IE 7 IT IS SO SLOW!
BUT ON IE 8 IT IS SO SLOW!
BUT ON IE 9 IT IS SO SLOW!
DID ANYONE TEST ON IE / SURFACE?
DID ANYONE TEST ON IE / SURFACE? MOBILE?
DID ANYONE TEST ON IE / SURFACE?

MOBILE?

CHEAP ANDROID DEVICES?
browsers day to day job
too many HTTP requests
> 2400 DOM elements

Awesome?
too many widgets
use large HTML chunks
use large HTML chunks
too many widgets ain’t good
WHY?
in GWT a Widget is a JS thing holding a DOM thing
CREATE CUSTOM WIDGETS
CREATE CUSTOM EVENTS

- don’t extend SimplePanel
- don’t extend VerticalPanel
- don’t extend FlextTable

extend composite!!!
the new native! do it in CSS…

```css
@-webkit-keyframes redPulse {
    from {
        box-shadow: 0px 0px 2px #ff0033;
    }
    50% {
        box-shadow: 0px 0px 10px #ff0033;
    }
    to {
        box-shadow: 0px 0px 2px #ff0033;
    }
}
```
use LayoutPanel
GWT Development
Basics
Structuring the UI
Talking to the server
SINGLE PAGE APPLICATION
NAVIGATION?
some action

mainPanel.setWidget(aWidget);

mainPanel.setWidget(bWidget);

„View A“

„View B“

„just do it“ pattern
hard to maintain
history
management
from day one!
back button and refresh as a feature
(not a catastrophe)
Keep It Stupid Simple

- use PLACES framework for main level navigation
- if you really need to, nest activities for a second level. try not to.
- use dialogs for user input, showing data. dialogs are easily reused.
top menue bound to places framework
switching between places with fade in and out
teach user to wait until application is ready again
gives us enough time to load the required content
300ms out

enough time

500ms in
Components inside of „activity“ fire non public, CUSTOM events
datepicker sends WEEK selected event
BOOKMARKABLE

presenter may goto itself, view may be cached
## Hashtag!
STATELESS VIEW

URL contains EVERYTHING needed to rebuild view

user hits reload

GWT apps starts, activity gets fired

view is back again

presenter loads data from server
some actions don‘t require PLACE navigation at all
use POPUPS/DIALOGS to stay ABOVE navigation
let POPUPS/DIALOGS move slowly into view
pin POPUPS to one side of the window
Don’t move your user away from his „PLACE“ unless you have to.

Search DIALOG slides in from right side, stays on TOP
Navigation should not hurt

- The application shown uses only 3 levels of navigation, DOES NOT NEED MORE
- PLACES used for bookmarkable entry points/ back button navigation consistency
- Activities should be STATELESS, to survive page reloads
- Learn from OTHERS, lookout for hashtags...
Once upon a time, a young good designer did a good looking design...

- he will be using photoshop or dreamweaver
- he will not use the software
- he will not build the software
- he will not maintain the software
BEFORE YOU ADD THE LOGO TO THE TOP

how many pixels do your USERS have?

the designer or marketing guy using Photoshop is probably sitting in front of an awesome 27" retina display
GWT Development
Basics
Structuring the UI
Talking to the server
GWT-RPC is a good solution if handled with care

GWT-RPC binds many methods into one interface

SomeResult someMethodName(SomeParameter spo)

Interface Versioning is a monstrous thing
SomeResult  someMethodName ( Type1  param1, Type2  param2, … )

this will be an object

introduce parameter object…
SomeResult  someMethodName ( SomeParameter spo )

this is still an object

so this will be an object too…
the method names bind the requests to the result

SomeResult someMethodName ( SomeParameter spo )
USING GENERICS FOR TYPESAFETY,
GET RID OF METHODS AND INTERFACES
nice to have:
one method to execute them all

R execute(A action);

typesafety is gone?
now we just have one interface with one method

```java
<A extends Action<R>, R extends Result> R execute(A action);
```

typesafety all the way
command pattern
GOF Pattern
commonly used in Rich Clients
someAction

EXECUTE

someActionHandler

someResult
public class TerminLoadAction implements Action<DataResult<TerminData>> {
    private String terminId;
    public TerminLoadAction(String terminId) {
        this.terminId = terminId;
    }
    public String getTerminId() {
        return terminId;
    }
}

public class DataResult<DATA extends Data> implements Result {
    private DATA data;
    public DataResult(DATA data) {
        this.data = data;
    }
    public void setData(DATA data) {
        this.data = data;
    }
    public DATA getData() {
        return data;
    }
}
dispatch.execute(
    new TerminLoadAction(terminId),
    new AsyncCallback<DataResult<TerminData>>() {
        @Override
        public void onFailure(Throwable caught) {
        }

        @Override
        public void onSuccess(DataResult<TerminData> result) {
        }
    });

<A extends Action<R>, R extends Result>
void execute(A action, AsyncCallback<R> callback)
public interface ActionHandler
    <A extends Action<R>, R extends Result> {
    
    Class<A> getActionType();

    R execute(
        A action,
        ExecutionContext context)
        throws DispatchException;
    
}
Server side

```java
@ActionHandlerBean
@Transactional
public final class TerminDataLoadHandler
    implements ActionHandler<TerminLoadAction, DataResult<TerminData>> {

    @Autowired
    private TerminDAO terminDao;

    @Override
    public DataResult<TerminData> execute(TerminLoadAction action, ExecutionContext context)
        throws DispatchException {
        TerminBean termin = ...
        TerminData data = ...
        return new DataResult<TerminData>(data);
    }

    @Override
    public Class<TerminLoadAction> getActionType() {
        return TerminLoadAction.class;
    }

    ...
```
interface
versioning
hell?
public interface SomeNiceService extends RemoteService {
    String someService(String param);
    String someServiceV2(String param);
    String someServiceV3(String param);
}

easy way

public interface SomeNiceService extends RemoteService {
    String someService(String param);
}

public interface SomeNiceServiceV2 extends RemoteService {
    String someService(String param);
}

public interface SomeNiceServiceV3 extends RemoteService {
    String someService(String param);
}

right way?

maintainability?
different versions can coexist!

someAction

someActionV2

someResult

multiple versions

someActionHandler

someActionHandlerV2

same result
multiple versions

different results

someAction
someActionV2
someResult
someResultV2

someActionHandler
someActionHandlerV2
why batch?
do it

then do this

finally do that

done
do it, then this, finally that

done
• one batch call is better than 10 single calls
• less data
• less roundtrip latency
• avoid connection bottleneck
• ensure server side execution order
• less roundtrips
batching can be manual or automatic

server executes actions in given order
automatic batching?
IDLE

Scheduler.scheduleEntry(…)

GWT code execution

Scheduler.scheduleFinally(…)

browser event loop

Scheduler.scheduleDeferred(…)

Diagram showing the flow of a system with a focus on GWT code execution within the browser event loop.
IDLE

Scheduler.scheduleEntry(…)

GWT code execution

Scheduler.scheduleFinally(…)

Scheduler.scheduleEntry(…)

GWT code execution

Scheduler.scheduleFinally(…)

collect commands
  cmd 1
  cmd 2
  cmd 3
  cmd …

fire batch command
BATCH EXECUTION

ENABLES FINE GRAINED COMMANDS AND REUSE

+ 

ENABLES ORDERING ACTIONS FOR EXECUTION ON THE SERVER
toggleTerminMetadata

reloadTermin

BooleanResult

DataResult<Termin>
<table>
<thead>
<tr>
<th>Monat</th>
<th>#</th>
<th>€</th>
<th>€/h</th>
</tr>
</thead>
<tbody>
<tr>
<td>Apr '13</td>
<td>38</td>
<td>957 €</td>
<td>28,14 €</td>
</tr>
<tr>
<td>Mai '13</td>
<td>31</td>
<td>1,128 €</td>
<td>38,23 €</td>
</tr>
<tr>
<td>Jun '13</td>
<td>2</td>
<td>80 €</td>
<td>40,00 €</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Termin</th>
<th>Patient</th>
<th>ReNr.</th>
<th>#</th>
<th>€</th>
</tr>
</thead>
<tbody>
<tr>
<td>01.05.13 09:00 (45m)</td>
<td>Behandlung</td>
<td>--</td>
<td>2</td>
<td>5,00</td>
</tr>
<tr>
<td>01.05.13 10:00 (45m)</td>
<td>Behandlung</td>
<td>--</td>
<td>2</td>
<td>5,00</td>
</tr>
<tr>
<td>01.05.13 14:00 (60m)</td>
<td>Behandlung</td>
<td>--</td>
<td>3</td>
<td>40,00</td>
</tr>
</tbody>
</table>

- toggleTerminMetadata
- loadMonthStats
- loadMonthTermine
- BooleanResult
- DataResult<MonthStats>
- DataListResult<Termin>
+ caching
+ reauth when session expired
+ exception handling
Thanks!

- GWT Development
- Basics
- Structuring the UI
- Talking to the server