Recording http traffic so that CaptainCasa can replay

The CaptainCasa server environment comes with a special feature that is inevitably important for solving complex bugs within the CaptainCasa frontend processing – that (as usual...) only happen inside your own environment and cannot be reproduced at CaptainCasa side.

There is a certain recording of all http traffic that is exchanged between the browser and the server. The recorded files can then be sent to CaptainCasa so that CaptainCasa can replay the frontend – the requests are responded from the files that you recorder. The result: CaptainCasa can exactly replay your scenario.

Security Issues

The recorded information contains all the traffic between your browser and your server. If the content contains critical information (including passwords, business information, …) then this is logged as consequence – and readable as clear text. So only use the recording feature in demo systems with demo user accounts...! If in doubt about security, please contact CaptainCasa.

Configuration

Recording (and replaying) is done by a servlet filter which is part of the normal CaptainCasa server environment.

Register the filter in the web.xml as follows:

```
<filter>
  <filter-name>org.eclnt.jsfserver.util.CompressionFilter</filter-name>
  <filter-class>org.eclnt.jsfserver.util.CompressionFilter</filter-class>
</filter>
<filter>
  <filter-name>org.eclnt.jsfserver.util.ResponseLoggerFilter</filter-name>
  <filter-class>org.eclnt.jsfserver.util.ResponseLoggerFilter</filter-class>
</filter>
<filter>
  <filter-name>org.eclnt.jsfserver.util.SecurityFilter</filter-name>
  <filter-class>org.eclnt.jsfserver.util.SecurityFilter</filter-class>
</filter>
<filter>
  <filter-name>org.eclnt.jsfserver.util.ThreadingFilter</filter-name>
  <filter-class>org.eclnt.jsfserver.util.ThreadingFilter</filter-class>
</filter>

<filter-mapping>
  <filter-name>org.eclnt.jsfserver.util.CompressionFilter</filter-name>
  <url-pattern>*.jsp</url-pattern>
</filter-mapping>
<filter-mapping>
  <filter-name>org.eclnt.jsfserver.util.CompressionFilter</filter-name>
  <url-pattern>*.xml</url-pattern>
</filter-mapping>
<filter-mapping>
  <filter-name>org.eclnt.jsfserver.util.ResponseLoggerFilter</filter-name>
  <url-pattern>*</url-pattern>
</filter-mapping>
<filter-mapping>
  <filter-name>org.eclnt.jsfserver.util.SecurityFilter</filter-name>
  <url-pattern>*.jsp</url-pattern>
</filter-mapping>0
  <filter-mapping>
  <filter-name>org.eclnt.jsfserver.util.ThreadingFilter</filter-name>
  <url-pattern>*.jsp</url-pattern>
</filter-mapping>
```

Your filter configuration might look a little bit different (e.g. you might not have the security filter installed). Please install the filter “behind” the CompressionFilter and “in front of” the SecurityFilter and ThreadingFilter. Please note: the sequence of filters is defined by the sequence of filter-mapping-definitions – and NOT by the sequence of filter-definitions.

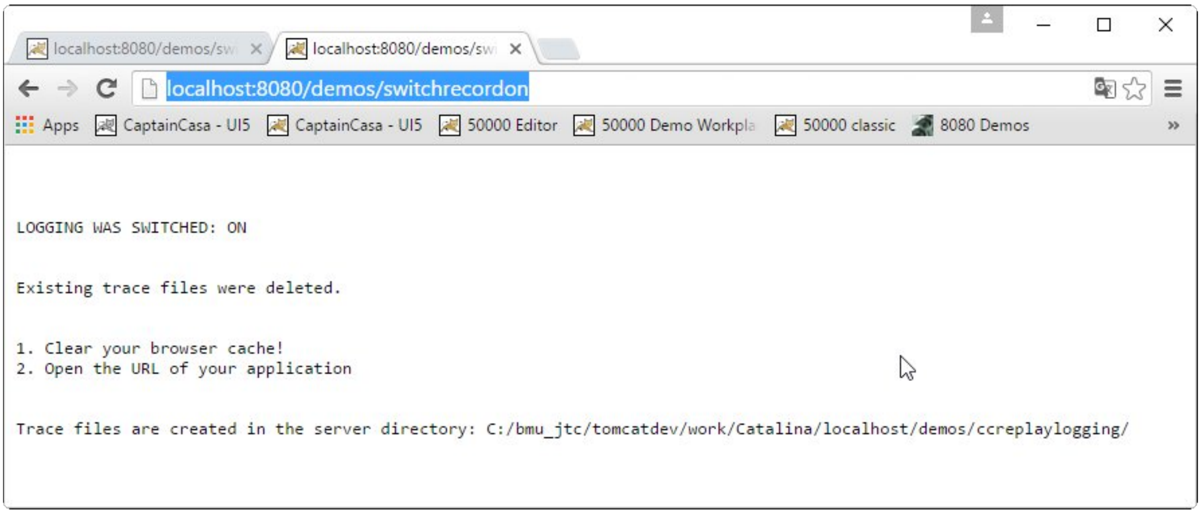
Please also pay attention: the filter MUST NOT be active in production scenarios! It is only to be used in development scenarios. (Otherwise any traffic is recorded!)

Recording

We explain the sequence of steps by using as example our demo workplace, which is available as “http://localhost:8080/demos/workplace.workplaceRisc.risc?ccstyle=defaultrisc” after installation.

Step 1 – Switch the recording to “on”

To do so you just need to call URL “http://localhost:8080/demos/switchrecordon”:



```
LOGGING WAS SWITCHED: ON


Existing trace files were deleted.


1. Clear your browser cache!
2. Open the URL of your application


Trace files are created in the server directory: C:/bmu_jtc/tomcatdev/work/Catalina/localhost/demos/ccreplaylogging/
```

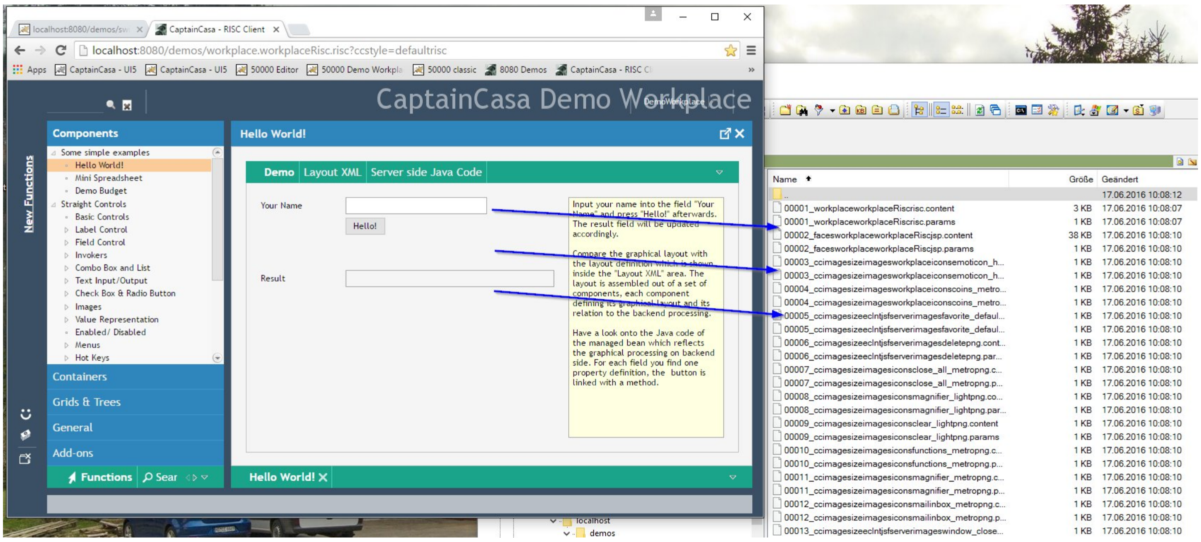
The text within the screen tells you the directory, in which the recorded files are placed.

Step 2 – Clear your browser cache

It is important that all requests are recorded that are required to run your page against the recorded log afterwards. So: clear your browser cache!

Step 3 – Run the URL of your application

Now you may start (e.g. in a second tab) the URL of your application just as normal. In the background on server side each request/response is recorded.



Input your name into the field "Your Name" and press "Hello!" afterwards. The result field will be updated accordingly.

Compare the graphical layout with the layout definition which is shown inside the "Layout XML" area. The layout is assembled out of a set of components, each component defining its graphical layout and its relation to the backend processing.

Have a look onto the Java code of the managed bean which reflects the graphical processing on backend side. For each field you find one property definition, the button is linked with a method.

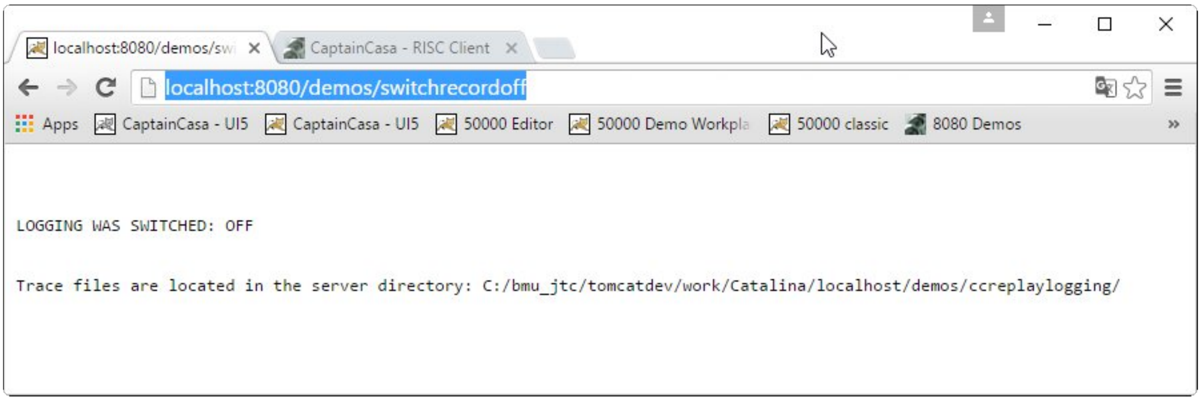
Name	Größe	Geändert
..		17.06.2016 10:08:12
00001_workplaceworkplaceRiscrisc.content	3 KB	17.06.2016 10:08:07
00001_workplaceworkplaceRiscrisc.params	1 KB	17.06.2016 10:08:07
00002_facesworkplaceworkplaceRiscjsp.content	38 KB	17.06.2016 10:08:10
00002_facesworkplaceworkplaceRiscjsp.params	1 KB	17.06.2016 10:08:10
00003_ccimagesizeimagesworkplaceiconsemoticon_h...	1 KB	17.06.2016 10:08:10
00003_ccimagesizeimagesworkplaceiconsemoticon_h...	1 KB	17.06.2016 10:08:10
00004_ccimagesizeimagesworkplaceiconscoins_metro...	1 KB	17.06.2016 10:08:10
00004_ccimagesizeimagesworkplaceiconscoins_metro...	1 KB	17.06.2016 10:08:10
00005_ccimagesizeeclntjsfserverimagesfavorite_defaul...	1 KB	17.06.2016 10:08:10
00005_ccimagesizeeclntjsfserverimagesfavorite_defaul...	1 KB	17.06.2016 10:08:10
00006_ccimagesizeeclntjsfserverimagesdeletepng.cont...	1 KB	17.06.2016 10:08:10
00006_ccimagesizeeclntjsfserverimagesdeletepng.par...	1 KB	17.06.2016 10:08:10
00007_ccimagesizeimagesiconsclose_all_metropng.c...	1 KB	17.06.2016 10:08:10
00007_ccimagesizeimagesiconsclose_all_metropng.p...	1 KB	17.06.2016 10:08:10
00008_ccimagesizeimagesiconsmagnifier_lightpng.co...	1 KB	17.06.2016 10:08:10
00008_ccimagesizeimagesiconsmagnifier_lightpng.par...	1 KB	17.06.2016 10:08:10
00009_ccimagesizeimagesiconsclear_lightpng.content	1 KB	17.06.2016 10:08:10
00009_ccimagesizeimagesiconsclear_lightpng.params	1 KB	17.06.2016 10:08:10
00010_ccimagesizeimagesiconsfunctions_metropng.c...	1 KB	17.06.2016 10:08:10
00010_ccimagesizeimagesiconsfunctions_metropng.p...	1 KB	17.06.2016 10:08:10
00011_ccimagesizeimagesiconsmagnifier_metropng.c...	1 KB	17.06.2016 10:08:10
00011_ccimagesizeimagesiconsmagnifier_metropng.p...	1 KB	17.06.2016 10:08:10
00012_ccimagesizeimagesiconsmailinbox_metropng.c...	1 KB	17.06.2016 10:08:10
00012_ccimagesizeimagesiconsmailinbox_metropng.p...	1 KB	17.06.2016 10:08:10
00013_ccimagesizeeclntjsfserverimageswindow_close...	1 KB	17.06.2016 10:08:10

Per request/response two files are written:

- The content of the response (".content")
- Parameters of the response (".params")

Step 4 – Switch the recording to "off"

Call URL "http://localhost:8080/demos/switchrecordoff":



```
LOGGING WAS SWITCHED: OFF

Trace files are located in the server directory: C:/bmu_jtc/tomcatdev/work/Catalina/localhost/demos/ccreplaylogging/
```

Step 5 – Send the recorded information to CaptainCasa

Now you may zip all of the recorded files to one file – and send it to CaptainCasa – together with the start-URL that you used.

...you want to replay on your own?

Well, replaying typically is done on CaptainCasa side, but you may want to do it for any reason on your own.

Just call URL "http://localhost:8080/demos/switchreplayon" and now the server will not work normally anymore but will fulfill all request by finding responses in the recorded files.

When calling your application URL "http://localhost:8080/demos/workplace.workplaceRisc.risc?ccstyle=defaultrisc" then all responses are coming "immediately" from the file system.

Finally switch the replay mode off by calling “http://localhost:8080/demos/switchreplayoff”.