Communication between Frames

You already saw some methods in the previous section enabling one frame to open pages in another frame and to refresh information of other frames.

This chapter covers the following topics:

- API inside the Adapter Class
- Pay Attention to Request Processing
- Session Management (I)
- Session Management (II)

API inside the Adapter Class

The following table shows a summary of functions that you can reach in your adapter class which inherits from `com.softwareag.cis.server.Adapter`. See the JavaDoc documentation for implementation details.

Method	Description
`openCISPageInTarget(...)`	Opens a certain Application Designer HTML page inside a certain frame which is identified by its target ID. There is a set of methods with different parameter notation. The default method just needs to know the page URL and the ID of the frame. Other methods expect more information, e.g. if you want to open the Application Designer page in a different subsession.
`refreshTarget(...)`	Refreshes the target’s frame content. This method is to be used if you want the target frame not to change its page but to update its content. In the example in the previous section, this method is used after having updated the right frame’s adapter on the server side.
`invokeMethodInTarget(...)`	Invokes a method in the target frame’s Application Designer HTML page. The call is triggered from the client - for example, imagine that a button supporting the method is pressed in the target frame’s Application Designer HTML page.
`sizeTarget(...)`	Manipulate the size of the target. Each target gets a certain size by the frame set definition on top of it (e.g. if the frame set definition has a sizing of "200,300,*", then the second frame has a size of "300". You can change the size of a target by using this method.

Pay Attention to Request Processing

Be aware of the request processing in the browser: only the page which sends a request (e.g. the left page in the example) is the active page and will process the response. All other pages living in neighboring frames are by default not affected.

Consequence: if you want to change or refresh these pages, you have to explicitly do so using the API presented in one of the previous sections.

Important:
The adapter that processes the request is the one to call the API methods.

Session Management (I)

Maybe you have already tried to build multi frame pages on your own, using HTML framesets:

```
...
...
<frameset cols="*,*">
  <frame src="../servlet/StartCISPage?PAGEURL=/project/left.html">
  <frame src="../servlet/StartCISPage?PAGEURL=/project/right.html">
</frameset>
...
...
```

If so, you will have seen that in each of the frames, the Application Designer page will be opened correctly. However, both pages are running in independent sessions (not subsessions).

Opening the same pages using Application Designer’s MF* controls (MFFRAMESET, MFCISPAGE) will keep both pages inisde the same session and subsession.

Note:
Details on session management are provided in the section *Session Management*.

Session Management (II)

When communicating between frames, e.g. by using the method `Adapter.openCISPageInTarget()`, the default is that the page that is opened in another target will be opened in the same session/subsession as the one that initiated the frame communication. Session ID and subsession ID are taken over by default.

There are certain variants of the `openCISPageInTarget()` method that allow to control the management of a subsession in a more fine granular way: you may pass as parameter the ID of the subsession in which a page should be opened in another page; i.e. you can explicitly decouple the other frame’s subsession from your own one.

The workplace that comes with Application Designer makes use of this: every time you open a content window, this content is managed in its own subsession, being decoupled from the workplace’s subsessions and being decoupled from other content pages’ subsessions.

Use these functions with care: typically all application adapters should run in one subsession, and only an "outside function" (such as the workplace management) should take care of starting various contents in various subsessions.