Table of Contents

Introduction

In January 2013 some colleagues, providers of education services, and I were discussing the advantages and disadvantages of running both a Moodle and a SmartCloud experience in parallel. Moodle is an open-source platform widely used in the education space to support curriculum design, course structure and it promotes interaction with students in the context of their formal class membership. Although SmartCloud has been used to support education processes, including remote teaching, file sharing, and collaborate learning, the adoption of Moodle is already in place. Education establishments are therefore already focused on running their teaching processes through Moodle and decision has impeded the adoption of SmartCloud in some education contexts.

There are, of course, many benefits of SmartCloud over Moodle, for example:

Cross-faculty and cross-school access of content
Enterprise authentication (no VPN required)
An ability to offer web-based lessons and meetings, and cross-platform application chat

Seamless integration across tools and platforms such as Microsoft Office™, Microsoft Outlook™, and IBM Notes™.

One approach to getting the best of both worlds is to combine aspects of Moodle's and IBM's open source approaches. That is, education establishments can still primarily interact through Moodle, but have access to SmartCloud content and functionality.

IBM is active in the open-source arena, and you can download IBM's Social Business Toolkit SDK (SBTK), as well as a number of other open-source projects from www.openntf.org. The SDK is intended for web and Java developers and is continually expanded to accommodate feedback from testers and users. This has resulted in an ever-evolving set of APIs accessible from Java, Javascript, or straight HTML. This SDK makes it possible to provide access to Connections and SmartCloud from many environments. In this particular instance a fairly pervasive tool used by staff and students worldwide. Because the SDK toolkit can be hosted from many environments (Tomcat, WebSphere, Portal, for example), adopters of this approach have access to a very wide choice of architectural patterns and authentication approaches. As of June 2013, Moodle has a user base of over 83,000 registered and verified sites and almost 71 million users. With minimal effort, there could potentially be an additional 71 million users of Connections/SmartCloud.

While many universities see value in social or collaborative platforms, for proper adoption of these, ideas need to support their existing educational processes such as creation and delivery of course material, as well as processes used by students such as accessing and sharing content from very different platforms (Microsoft Office, Microsoft Windows Explorer™, and publically available social networks). This is where the potential conflict lies. With only so much time at one's disposal, how can I as a user embed access to complementary services through a platform that is already part of my 'day job'?

The approach taken in this article

I will start with an empty un-themed Moodle environment, and walk through the integration approach. I will then outline some code in Moodle, some code in the SBTK, and how we integrate the two. I will then describe how to customise the look and feel of the Connections content to suit the Educator's local environment.

There are a couple of ways to provide access to "Connections" content. I will also show how this can be configured to use both (even simultaneously). For demonstration purposes, I will use two publically-available Connections environments. The first is Greenhouse, which is the equivalent of an on-premises Connections environment but accessible in the cloud. Access to this is free and could be considered a testing area for many IBM Collaboration offerings. The second is the commercially available IBM SmartCloud for Social Business with guest and trial options. Links to both of these can be found in the references section at the end of the article.

You will be shown how to:

1) Create a Moodle block to access to Connections files.
2) Create user-defined Moodle profile fields to customise your experience.
3) Connect to both Connections on premises and Connections in the cloud.
4) Theme your SBTK blocks (using CSS)

Assumptions

This is an introduction to integration options between Moodle and Connections, so I will only cover the basics. In summary:

I will use Moodle 2.x , however you can use any Moodle version, subject to Moodle coding constraints.

I will use the included SBTK 'Playground' to host our SBTK snippet code with the included Tomcat server. You could also deploy this against another Application server, but this is beyond the scope of this article.

I will not focus on scalability, resilience, or high availability. While these are important topics for a deployment involving many thousands of students it is also beyond the scope of this article.

Both Greenhouse and SmartCloud provide a Connections environment, so unless I refer to a specific instance or particular functionality, I will use Connections to refer to what is generally available in both. Otherwise, I will refer specifically to Connections on-premises or SmartCloud for Social Business.

You will need accounts for Greenhouse and SmartCloud for Social Business. These are easy to get so if you do not have them, get them now and come back to this part in the article when you have access.

What does the architecture look like?

Figure 1 shows a simplified view of the proposed architecture. In summary, you add a custom Moodle block to your existing Moodle environment that exposes content contained in Connections. To access this content, I use the Social Business toolkit, which includes a standard Tomcat server.

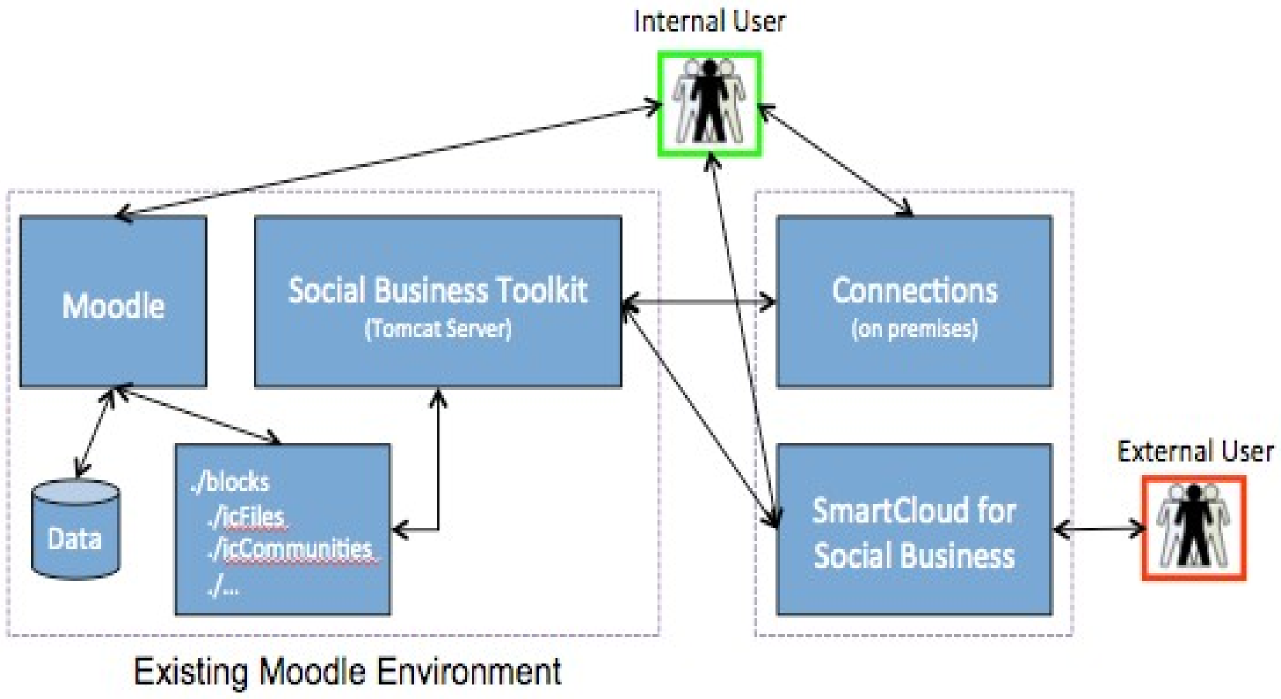


Figure 1: A simplified architecture showing Moodle and Connections

The primary interface to Connections content is the Tomcat server, not Moodle. Therefore I use a Moodle block that points to routines contained within the Social Business Toolkit, which is hosted inside Tomcat. I do however, have some flexibility around how this content will be displayed even though Moodle manages the display interface.

The installation procedure(s)

Installing Moodle

I assume that Moodle is already installed in your environment. If you do not have Moodle installed, there are a number of good guides on how to do this. This article assumes that you are adding capabilities to your Moodle installation without affecting anything you already have installed.

Make a note of your Moodle install directory here:

MOODLE_HOME:

____

Installing the Social Business SDK

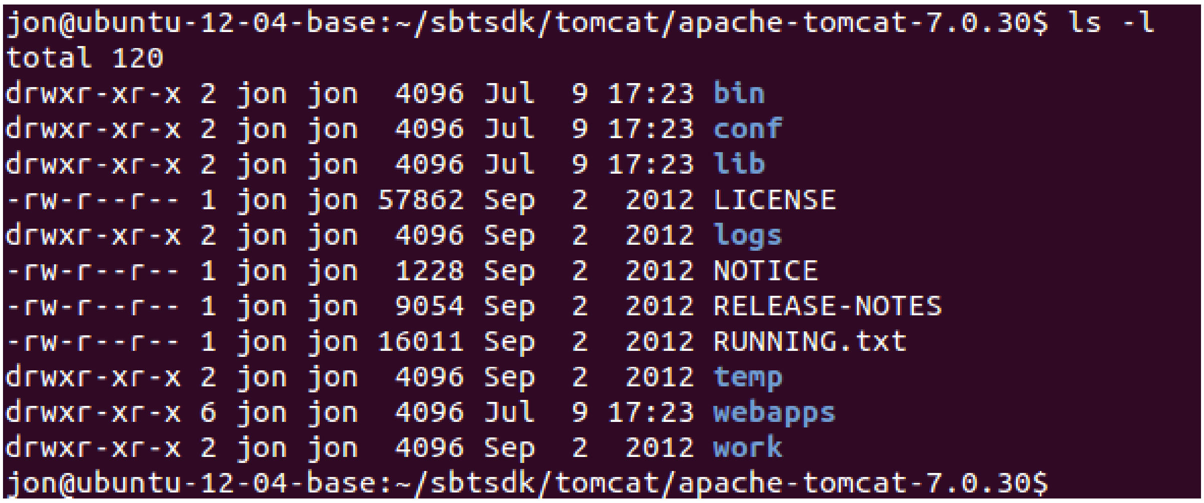
Download the SBTK zip file from http://ibmsbt.openntf.org. Extract the zip file and make a note of its location here:

SBTK_HOME:

____

The SBTK already contains a Tomcat server, which requires a working Java Runtime Environment (JRE). Ensure you have one installed.

Change directory to **$SBTK_HOME/tomcat**, which contains a zip file of the tomcat server. The file will be called something like apache-tomcat-7.0.30-sbt.zip. Unzip that file. You will see a list of directories like this:



```
jon@ubuntu-12-04-base:~/sbtsdk/tomcat/apache-tomcat-7.0.30$ ls -l
total 120
drwxr-xr-x 2 jon jon  4096 Jul  9 17:23 bin
drwxr-xr-x 2 jon jon  4096 Jul  9 17:23 conf
drwxr-xr-x 2 jon jon  4096 Jul  9 17:23 lib
-rw-r--r-- 1 jon jon 57862 Sep  2  2012 LICENSE
drwxr-xr-x 2 jon jon  4096 Sep  2  2012 logs
-rw-r--r-- 1 jon jon  1228 Sep  2  2012 NOTICE
-rw-r--r-- 1 jon jon  9054 Sep  2  2012 RELEASE-NOTES
-rw-r--r-- 1 jon jon 16011 Sep  2  2012 RUNNING.txt
drwxr-xr-x 2 jon jon  4096 Sep  2  2012 temp
drwxr-xr-x 6 jon jon  4096 Jul  9 17:23 webapps
drwxr-xr-x 2 jon jon  4096 Sep  2  2012 work
jon@ubuntu-12-04-base:~/sbtsdk/tomcat/apache-tomcat-7.0.30$
```

Figure 2: The contents of the apache-tomcat directory

The three directories of most interest are:

1. **./bin** – contains the programs to start and stop the Tomcat server.
2. **./conf** – contains the files required to configure Connections end points.
3. **./webapps** – contains sample code, this is also where you store SBTK snippets.

Point the SBTK at Connections

Use Greenhouse to point the SBTK at a Connections repository, .
Beneath the **conf** directory, edit the **sbt.properties** file.

Underneath the section headed **Connections Basic Endpoint Parameters** modify the **connections.url** parameter to point to the Greenhouse server.

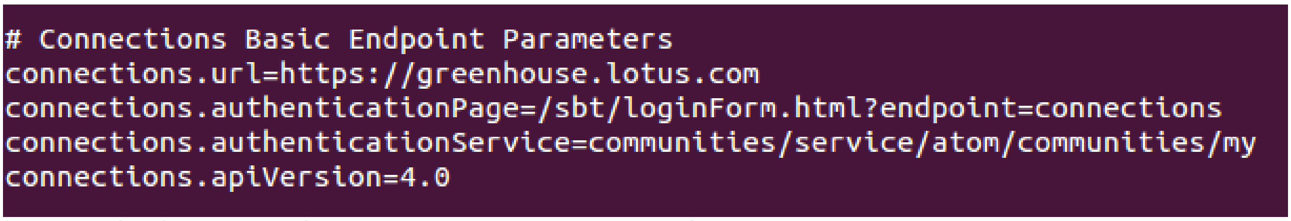
Change:

connections.url=https://qs.renovations.com:444

to

connections.url=https://greenhouse.lotus.com

which will result in output like this:

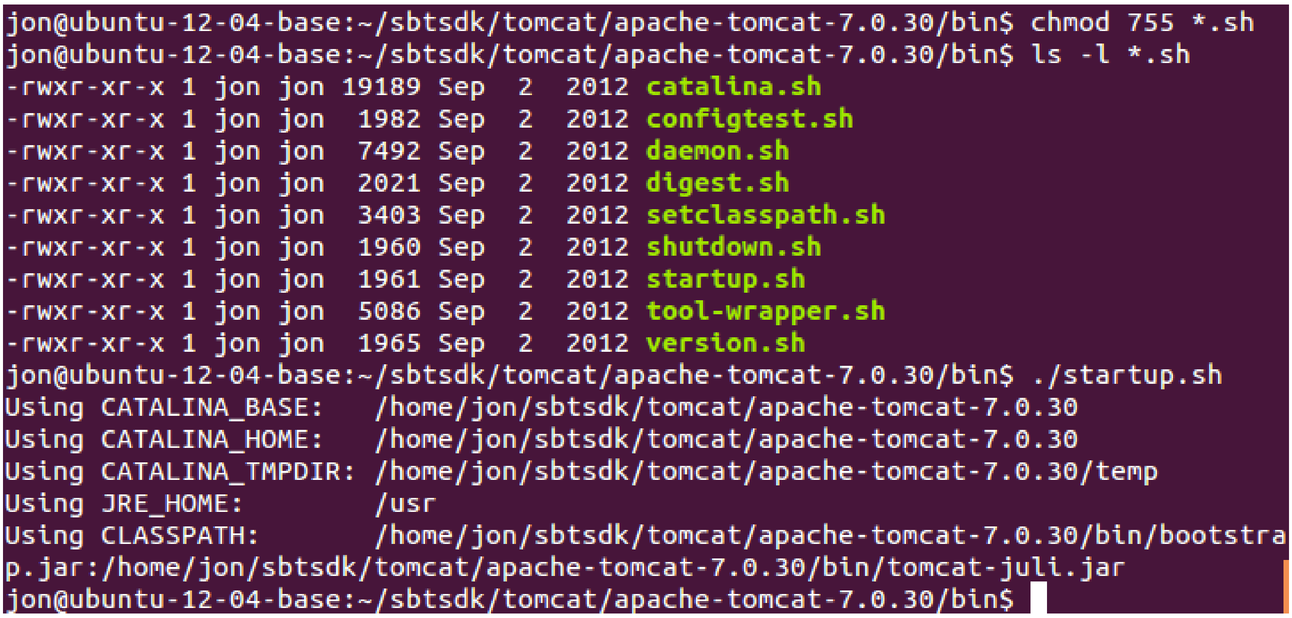


```
# Connections Basic Endpoint Parameters
connections.url=https://greenhouse.lotus.com
connections.authenticationPage=/sbt/loginForm.html?endpoint=connections
connections.authenticationService=communities/service/atom/communities/my
connections.apiVersion=4.0
```

Figure 3: Configuring the Connections Endpoint to point at Greenhouse.

Start Tomcat

Start the Tomcat server. The shell scripts within **bin** are not executable by default, so change their mode to 755, and start Tomcat by running **startup.sh** as follows:

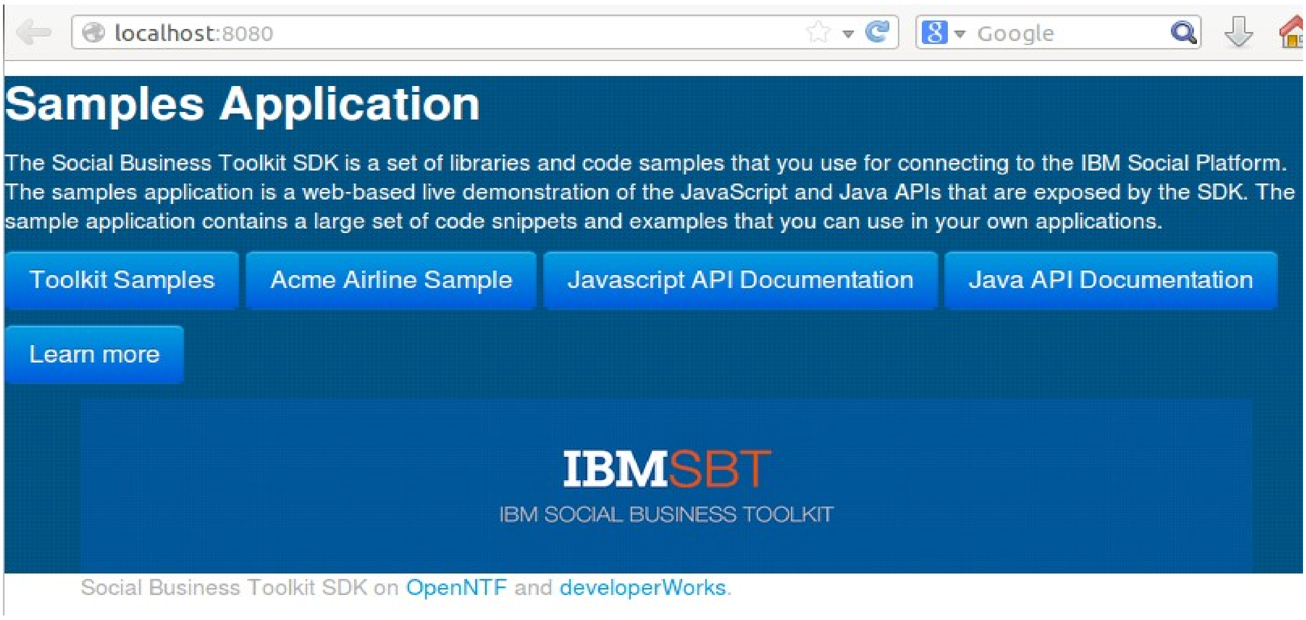


```
jon@ubuntu-12-04-base:~/sbtsdk/tomcat/apache-tomcat-7.0.30/bin$ chmod 755 *.sh
jon@ubuntu-12-04-base:~/sbtsdk/tomcat/apache-tomcat-7.0.30/bin$ ls -l *.sh
-rwxr-xr-x 1 jon jon 19189 Sep  2  2012 catalina.sh
-rwxr-xr-x 1 jon jon  1982 Sep  2  2012 configtest.sh
-rwxr-xr-x 1 jon jon  7492 Sep  2  2012 daemon.sh
-rwxr-xr-x 1 jon jon  2021 Sep  2  2012 digest.sh
-rwxr-xr-x 1 jon jon  3403 Sep  2  2012 setclasspath.sh
-rwxr-xr-x 1 jon jon  1960 Sep  2  2012 shutdown.sh
-rwxr-xr-x 1 jon jon  1961 Sep  2  2012 startup.sh
-rwxr-xr-x 1 jon jon  5086 Sep  2  2012 tool-wrapper.sh
-rwxr-xr-x 1 jon jon  1965 Sep  2  2012 version.sh
jon@ubuntu-12-04-base:~/sbtsdk/tomcat/apache-tomcat-7.0.30/bin$ ./startup.sh
Using CATALINA_BASE:   /home/jon/sbtsdk/tomcat/apache-tomcat-7.0.30
Using CATALINA_HOME:   /home/jon/sbtsdk/tomcat/apache-tomcat-7.0.30
Using CATALINA_TMPDIR: /home/jon/sbtsdk/tomcat/apache-tomcat-7.0.30/temp
Using JRE_HOME:        /usr
Using CLASSPATH:       /home/jon/sbtsdk/tomcat/apache-tomcat-7.0.30/bin/bootstra
p.jar:/home/jon/sbtsdk/tomcat/apache-tomcat-7.0.30/bin/tomcat-juli.jar
jon@ubuntu-12-04-base:~/sbtsdk/tomcat/apache-tomcat-7.0.30/bin$ 
```

Figure 4: Making the shell scripts executable and starting Tomcat

You will see the environment variables being set, and if your Java process is running, you will be able to go to http://localhost:8080 and see the Samples application page (Figure)

Working with the Samples Application



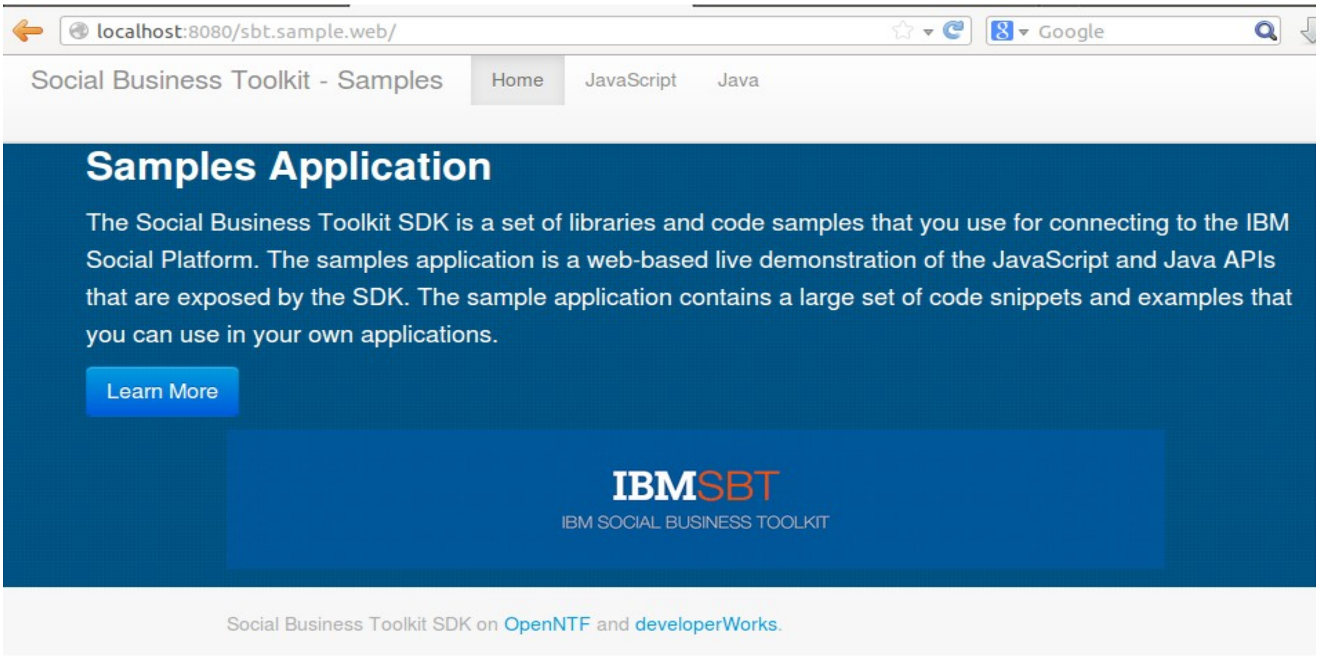
Samples Application

The Social Business Toolkit SDK is a set of libraries and code samples that you use for connecting to the IBM Social Platform. The samples application is a web-based live demonstration of the JavaScript and Java APIs that are exposed by the SDK. The sample application contains a large set of code snippets and examples that you can use in your own applications.

Social Business Toolkit SDK on OpenNTF and developerWorks.

Figure 5: The Samples application available from Tomcat

Click on the Toolkit Samples button, you will see the ‘Playground’, which allows you to look at and play with SBTK snippets in either JavaScript or Java.


localhost:8080/sbt.sample.web/
Google
Social Business Toolkit - Samples
Home
JavaScript
Java
Learn More
IBMSBT
IBM SOCIAL BUSINESS TOOLKIT

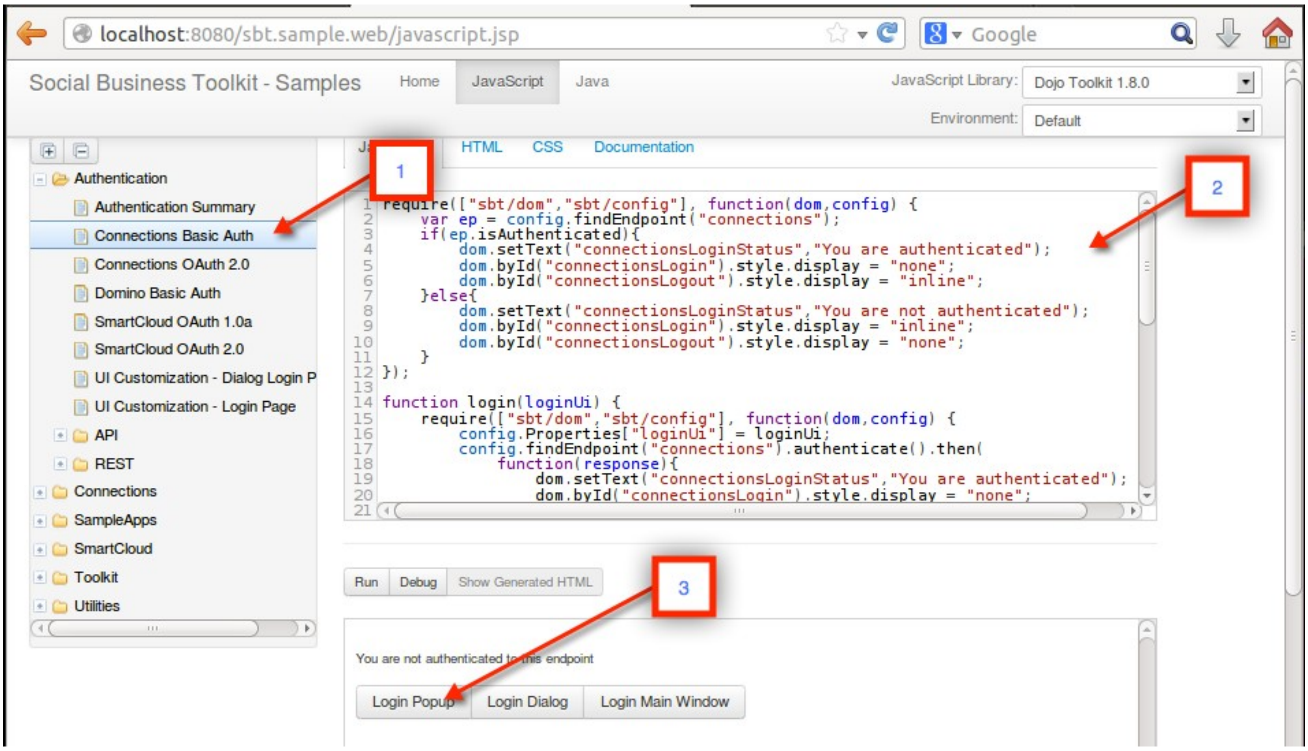
Samples Application

The Social Business Toolkit SDK is a set of libraries and code samples that you use for connecting to the IBM Social Platform. The samples application is a web-based live demonstration of the JavaScript and Java APIs that are exposed by the SDK. The sample application contains a large set of code snippets and examples that you can use in your own applications.

Social Business Toolkit SDK on OpenNTF and developerWorks.

Figure 6: The Toolkit Samples entry point.

Click on the word ‘Javascript’, to enter the playground. Open the Authentication entry in the tree on the left hand side and choose ‘Connections Basic Auth’, where you can now test connectivity to Greenhouse.



```
require(["sbt/dom","sbt/config"], function(dom,config) {
    var ep = config.findEndpoint("connections");
    if(ep.isAuthenticated){
        dom.setText("connectionsLoginStatus","You are authenticated");
        dom.byId("connectionsLogin").style.display = "none";
        dom.byId("connectionsLogout").style.display = "inline";
    }else{
        dom.setText("connectionsLoginStatus","You are not authenticated");
        dom.byId("connectionsLogin").style.display = "inline";
        dom.byId("connectionsLogout").style.display = "none";
    }
});

function login(loginUi) {
    require(["sbt/dom","sbt/config"], function(dom,config) {
        config.Properties["loginUi"] = loginUi;
        config.findEndpoint("connections").authenticate().then(
            function(response){
                dom.setText("connectionsLoginStatus","You are authenticated");
                dom.byId("connectionsLogin").style.display = "none";
```

You are not authenticated to this endpoint

Figure 7: The layout of the SBTK playgroun

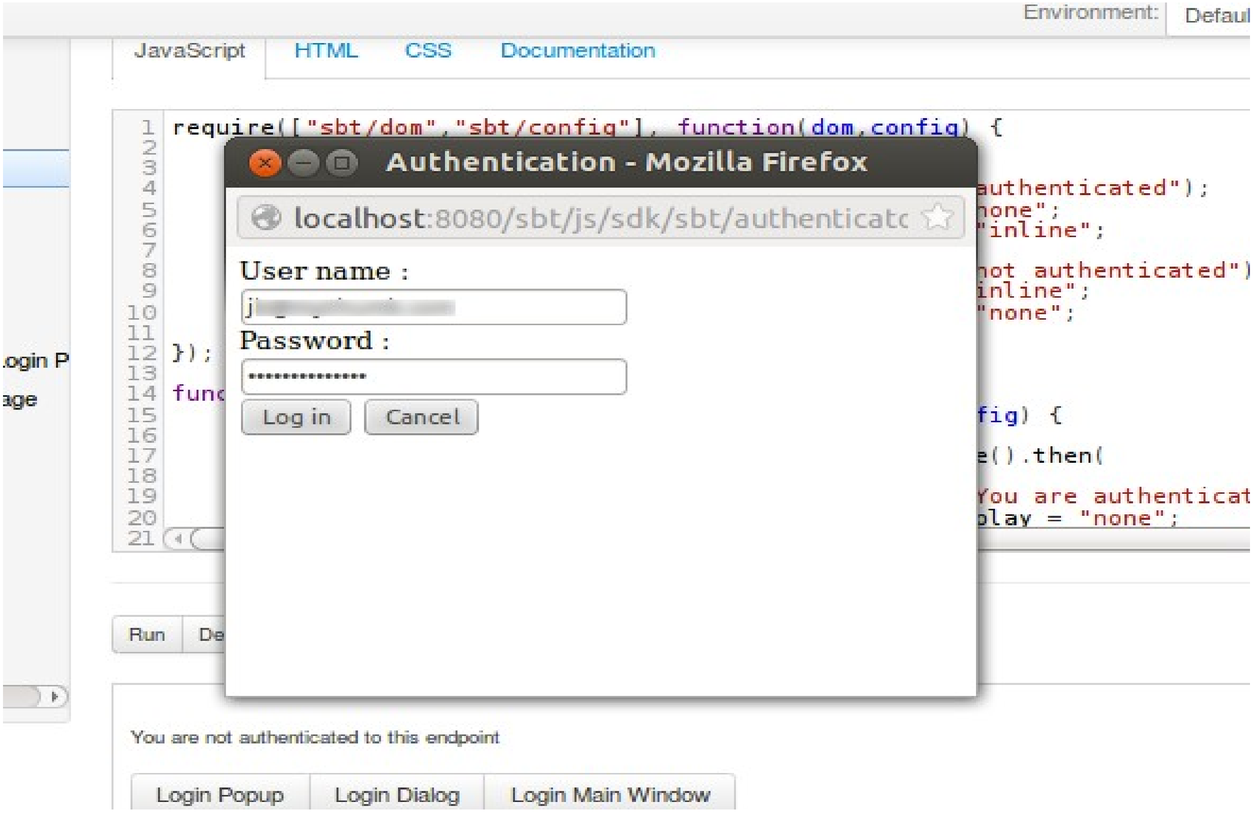
The three areas to highlight are:

(1) The snippet tree on the left hand side, which lets you choose snippets to play with.

(2) The actual snippet code in the top right half of the screen

(3) The output of the snippet code.

Choose '**Connections Basic Auth**', and then click on Login Popup, which will prompt for Greenhouse credentials.



```
require(["sbt/dom","sbt/config"], function(dom,config) {
                                                        authenticated");
                                                        none";
                                                        "inline";

                                                        not authenticated")
                                                        inline";
                                                        "none";

});

func
                                                        fig) {
                                                        e().then(
                                                        You are authenticat
                                                        play = "none";
```

Authentication - Mozilla Firefox

localhost:8080/sbt/js/sdk/sbt/authenticatc

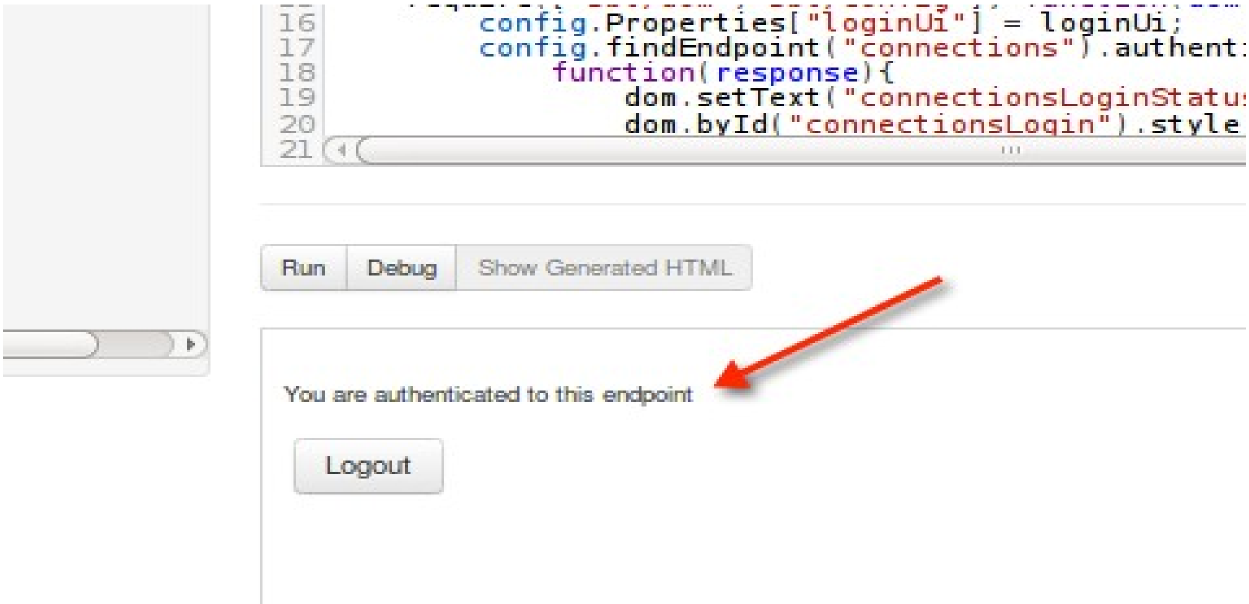
User name : [redacted]

Password : ••••••••••••••••

You are not authenticated to this endpoint

Figure 8: Testing Greenhouse connectivity.

You will see the status in the snippet code output section change to a message showing that you are now authenticated (Figure).

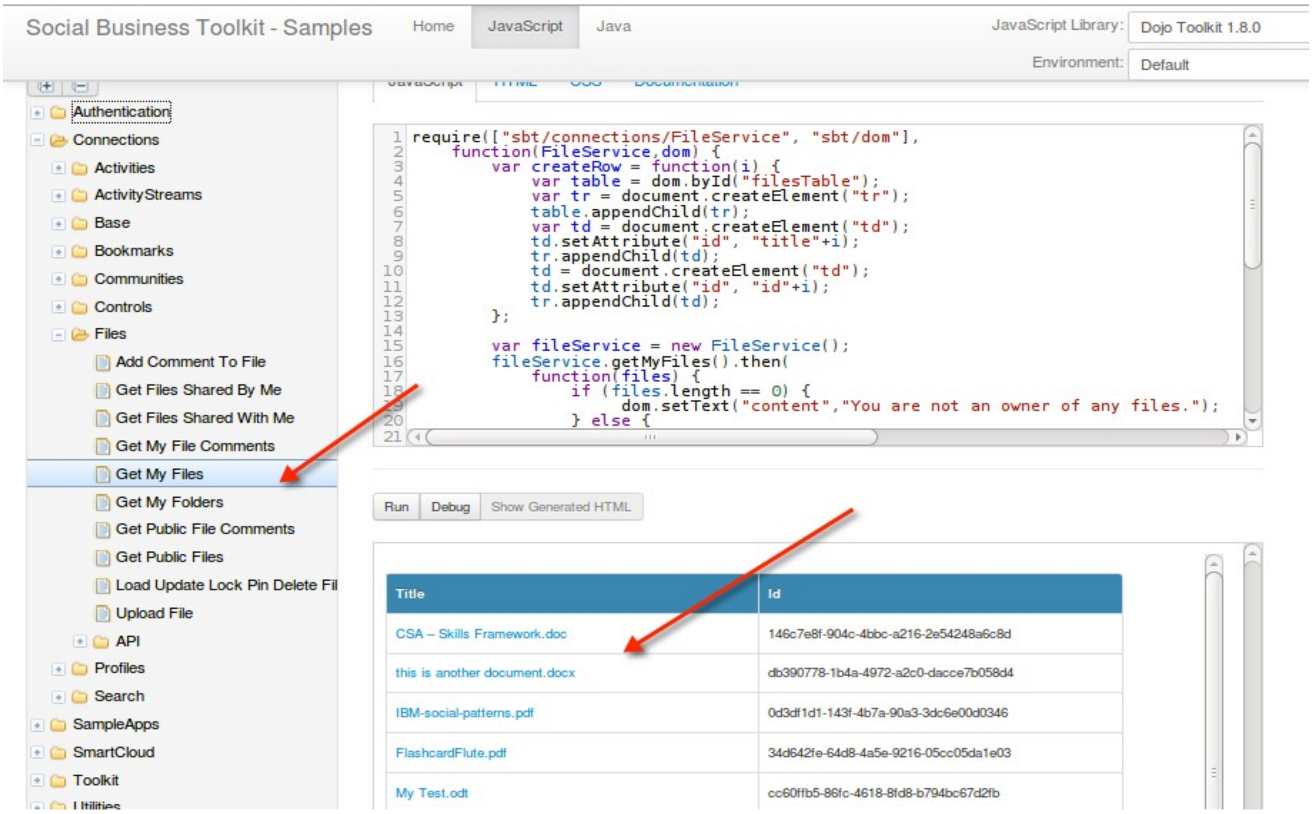


```
        config.Properties["loginUi"] = loginUi;
        config.findEndpoint("connections").authent
            function(response){
                dom.setText("connectionsLoginStatu
                dom.byId("connectionsLogin").style
```

You are authenticated to this endpoint

Figure 9: A successful authentication.

Lastly, test that you can get a list of the files stored in Greenhouse. Choose the '**Connections**' main section on the left, and then beneath '**Files'**, choose '**Get My Files**'.



JavaScript Library: Dojo Toolkit 1.8.0

Environment: Default

```
require(["sbt/connections/FileService", "sbt/dom"],
    function(FileService,dom) {
        var createRow = function(i) {
            var table = dom.byId("filesTable");
            var tr = document.createElement("tr");
            table.appendChild(tr);
            var td = document.createElement("td");
            td.setAttribute("id", "title"+i);
            tr.appendChild(td);
            td = document.createElement("td");
            td.setAttribute("id", "id"+i);
            tr.appendChild(td);
        };

        var fileService = new FileService();
        fileService.getMyFiles().then(
            function(files) {
                if (files.length == 0) {
                    dom.setText("content","You are not an owner of any files.");
                } else {
```

Title	Id
CSA – Skills Framework.doc	146c7e8f-904c-4bbc-a216-2e54248a6c8d
this is another document.docx	db390778-1b4a-4972-a2c0-dacce7b058d4
IBM-social-patterns.pdf	0d3df1d1-143f-4b7a-90a3-3dc6e00d0346
FlashcardFlute.pdf	34d642fe-64d8-4a5e-9216-05cc05da1e03
My Test.odt	cc60ffb5-86fc-4618-8fd8-b794bc67d2fb

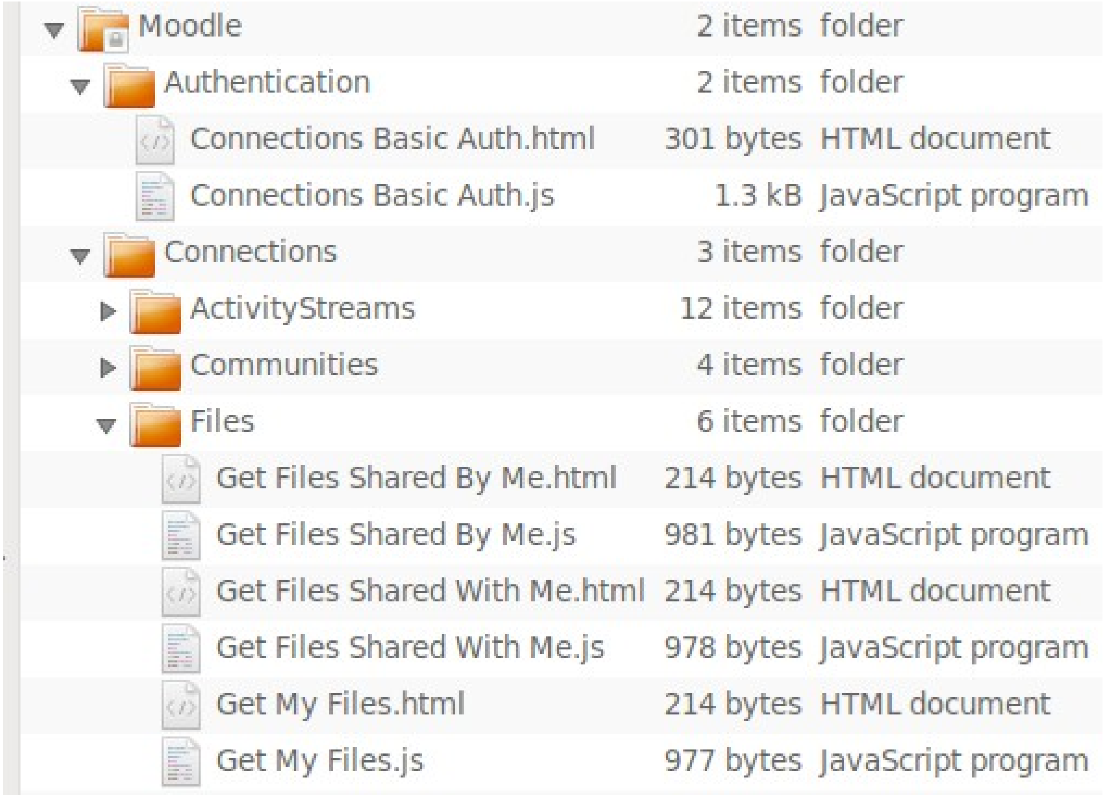
Figure 10 : Testing that we can see our Connections Files

The output is not well formatted but it does provide real data. The SBTK simply provides sample snippet code to show you how to access the APIs in the context of your Connections content, but in practice you would probably customise it to suit your own requirements, with your own style sheets. With this in mind, I have written my own snippet code, which I will add to the Sample snippet directory accessed directly from Moodle.

Adding additional SBTK Snippets

As you have seen, the SBTK comes with a number of sample snippets. Now add a couple more to give you something better to customise.

Extract the file **MoodleSnippets.zip** into the **$SBTK_HOME/tomcat/apache-tomcat-7.0.30/webapps/sbt.sample.web/samples/js** directory.
If you go back to http://localhost:8080/sbt.sample.web/javascript.jsp, it will now have a Moodle sub-directory, and a number of useful routines. I have taken the sample snippets and made them a little prettier. Each snippet has a Javascript routine as well as an optional HTML file, which I discuss in more detail later.



Moodle	2 items	folder
Authentication	2 items	folder
Connections Basic Auth.html	301 bytes	HTML document
Connections Basic Auth.js	1.3 kB	JavaScript program
Connections	3 items	folder
ActivityStreams	12 items	folder
Communities	4 items	folder
Files	6 items	folder
Get Files Shared By Me.html	214 bytes	HTML document
Get Files Shared By Me.js	981 bytes	JavaScript program
Get Files Shared With Me.html	214 bytes	HTML document
Get Files Shared With Me.js	978 bytes	JavaScript program
Get My Files.html	214 bytes	HTML document
Get My Files.js	977 bytes	JavaScript program

Figure 11: The contents of the Moodle snippets directory.

Building a Moodle block to call our SBTK routines

There are a number of ways of adding functionality into Moodle and we will use Blocks for the following reasons:

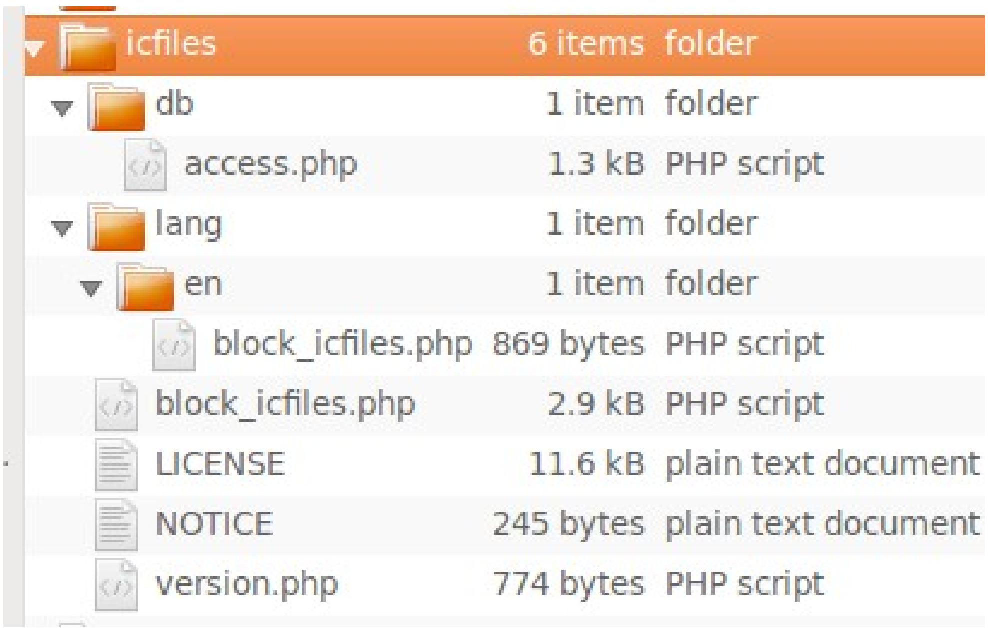
Existing Moodle environments do not need any customisation. A Block can simply be copied into the Moodle’s **blocks** directory and the administrator can enable it.

Individual Blocks can be added/enabled or removed/disabled in parallel, so the combination of Connections functionality you use is flexible.

Each Block can be customised in isolation. You may want Blocks that do very specific things for different users, or that look differently for different communities. For example, Blocks that access external (e.g. SmartCloud) content might obviously highlight this, say with a particular theme to let people know that this content is not ‘internal’.

Because there are many publically available guides to show how to create a Moodle Block, I will only explain what you need to know to get the basics of the Block working.

In Figure , you can see the constituent files from a block that I have called **icfiles**. The **db/access.php** file dictates what your block is allowed to do, and the **lang/en/block_icfiles.php** provides English specific descriptions for your Block that is visible in your Moodle installation. Here also is where you customise how the names of your blocks are referenced in a user-friendly manner.



icfiles	6 items	folder
db	1 item	folder
access.php	1.3 kB	PHP script
lang	1 item	folder
en	1 item	folder
block_icfiles.php	869 bytes	PHP script
block_icfiles.php	2.9 kB	PHP script
LICENSE	11.6 kB	plain text document
NOTICE	245 bytes	plain text document
version.php	774 bytes	PHP script

Figure 12: The Connections 'Files' block code (blocks/icfiles)

```
<?php

$capabilities = array(

    'block/icfiles:myaddinstance' => array(
        'captype' => 'write',
        'contextlevel' => CONTEXT_SYSTEM,
        'archetypes' => array(
            'user' => CAP_ALLOW
        ),

        'clonepermissionsfrom' => 'moodle/my:manageblocks'
    ),

    'block/icfiles:addinstance' => array(
        'riskbitmask' => RISK_SPAM | RISK_XSS,

        'captype' => 'write',
        'contextlevel' => CONTEXT_BLOCK,
        'archetypes' => array(
            'editingteacher' => CAP_ALLOW,
            'manager' => CAP_ALLOW
        ),

        'clonepermissionsfrom' => 'moodle/site:manageblocks'
    ),
);
```

Listing 1 : Access rights provided for the Connections Files Block (db/access.php).

```
<?php

$string['pluginname'] = 'Connections Files Block';
$string['icfiles'] = 'Connections Files';
$string['icfiles:addinstance'] = 'Add a new Connections Files block';
```

```
$string['icfiles:myaddinstance'] = 'Add a new Connections Files block to My Moodle page';
```

Listing 2: Language specific descriptions for your Block (lang/en/block_icfiles.php)

Let us look at the PHP block code and then I will summarise what it does.

```
    <script>
1           function setFilesIframeSource() {
                var theSelect = document.getElementById('url');
                var theIframe = document.getElementById('filesFrame');
                var theUrl;
                theUrl = theSelect.options[theSelect.selectedIndex].value;
                theIframe.src = theUrl;
            }
    </script>

    <?php

    require_once($CFG->dirroot.'/user/profile/lib.php');

    class block_icfiles extends block_base {
            public function init() {
2                   $this->title = get_string('icfiles', 'block_icfiles');
            }

            public function get_content() {

                    global $USER;           // Needed to access private field values

                    if ($this->content !== null) {
                            return $this->content;
                    }
                    $this->content          =  new stdClass;

            // Customise behaviour based on profile settings
            $myuser = new object();
            $myuser = profile_user_record($USER->id);
3           $ICSCEnv=$myuser->IBMICSC;

4           $this->content->text   "<form id='form1' method='post'>";

5           $this->content->text   .= "<select id='url' onchange='setFilesIframeSource()'>";
6           $this->content->text   .= "<option 
    value='https://localhost:8443/sbt.sample.web/javascriptPreview.jsp?
    snippet=Moodle_Connections_Files_Get_My_Files&jsLibId=dojo180&env=" . $ICSCEnv . "'>My 
    Files</option>";
            $this->content->text   .= "<option 
    value='https://localhost:8443/sbt.sample.web/javascriptPreview.jsp?
    snippet=Moodle_Connections_Files_Get_Files_Shared_With_Me&jsLibId=dojo180&env=" . $ICSCEnv . 
    "'>Shared With me</option>";
            $this->content->text   .= "<option 
    value='https://localhost:8443/sbt.sample.web/javascriptPreview.jsp?
    snippet=Moodle_Connections_Files_Get_Files_Shared_By_Me&jsLibId=dojo180&env=" . $ICSCEnv . 
    "'>Shared By me</option>";
            $this->content->text   .= "</select></form>";
7           $this->content->text   .= "</form>";
8
            $this->content->text   .= "<iframe id='filesFrame' 
    src='https://localhost:8443/sbt.sample.web/javascriptPreview.jsp?
    snippet=Moodle_Connections_Files_Get_My_Files&jsLibId=dojo180&env=" . $ICSCEnv . "' width='200' 
    style='border-style:none;'></iframe>";

            $this->content->footer = "";
            return $this->content;
        }

    }   // Closing bracket for the class definition
```

Listing 3: A very simple Moodle block

This is what each section does:

1 A Javascript function that forces a refresh when the drop-down value changes.

2 The Block header description is pulled from **lang/en/block_icfiles.php**

3 This variable allows you to selectively retrieve content from either a Connections or SmartCloud environment. I describe later how to create a user profile field.

4 The beginning of the HTML form containing my drop-down selector and the ordered list containing my files.

5 The drop-down with a call to the refresh routine on a change of value

6 The first or each of the available options. Note the following:

The value is the URL of the SBTK routine that needs to be called to populate the file list.

The URL contains a reference to the environment **(&env=…)** containing the files. Namely, Greenhouse or SmartCloud for Social Business.

7 The content is shown as the output of an iFrame. This has a number of implications. Firstly, any theming/skinning that Moodle controls is not passed to the iFrame content. Secondly, the Moodle HTTP session and the iFrame's Tomcat HTTP session are different, so even if you are logged into your Connections environment on a different Browser session, you need to provide a log in via Tomcat as well. We can remedy this using a single-sign-on approach via OAuth or a SAML login, but this is beyond the scope of this article.

8 This is the default output for the block (i.e., show My Files).

1able 1: Notes to describe the main Block's source code

This is what the block does:

By default, show files I have stored in my Connections repository.

Provide a drop-down selector where I can choose whether I see 'My Files', 'Files Shared with Me', or 'Files Shared by Me'.

Get me the content from my preferred Connections environment (on-premises Connections or SmartCloud for Social Business).

Analysing the SBTK Snippet

The Moodle Block selectively exposes the Connections capability to a Moodle user. However, the hard work of getting the content is done by the SBTK snippet. Let us walk through the snippet that gets My Files.

```
1  require(["sbt/connections/FileService", "sbt/dom"], function(FileService,dom) {
2      var fileService = new FileService();
3      fileService.getMyFiles({ ps: 100 }).then(
           function(files) {
4              if (files.length == 0) {
                   dom.setText("content","You are not an owner of any files.");
               } else {
5                  var olElement = document.createElement("ol");

                   document.getElementById("content").appendChild(olElement);

6                  for(var i=0; i<files.length; i++){
                       var strText="";
                       var strName = files[i].getTitle();
                       var strDisplay = strName.substring(0,25);

7                      if (strDisplay.length >= 25) {
                           strDisplay += "...";
```

```
                         }
8                        strText += "<li><a href='" + files[i].getDownloadUrl() + "' title='" +
    strName + "'>";
                         strText += strDisplay;
                         strText += "</a></li>"
                         olElement.innerHTML += strText;
                    }
                }
            },
9           function(error) {
                dom.setText("content", "Error code:" +  error.code + ", message:" + error.message);
            }
        );
    }
```

Listing 4: The SBTK snippet to get 'My Files'

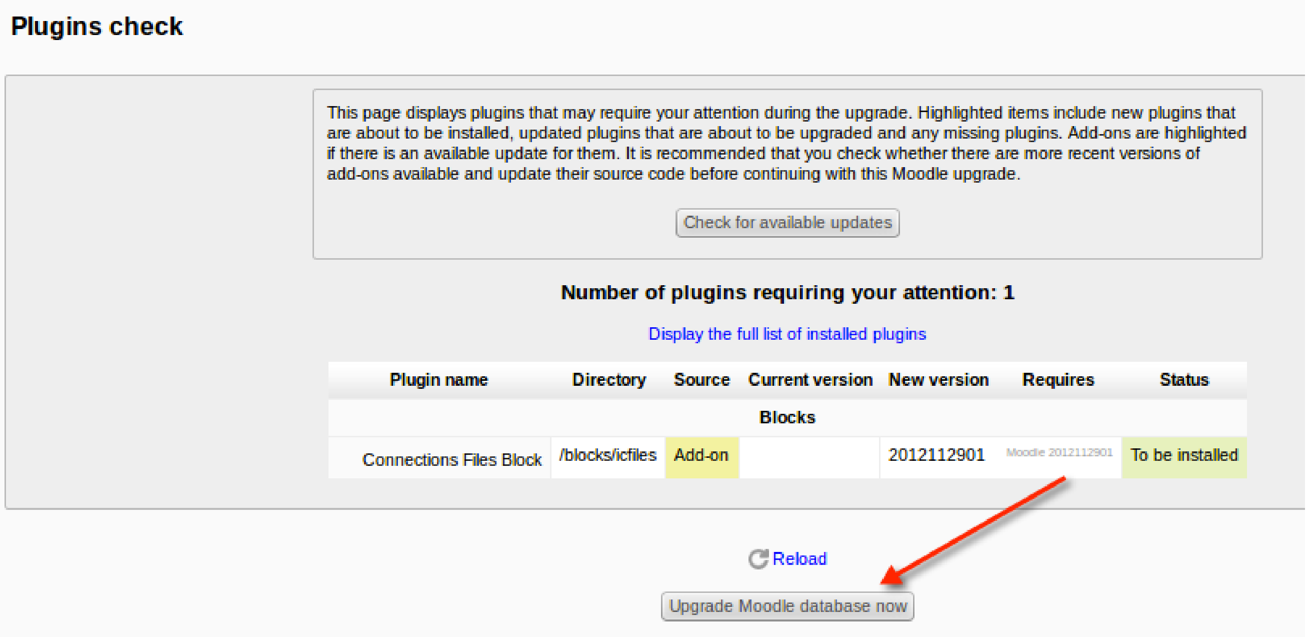
1 Load pre-requisite libraries for FileService
2 Define a variable to call those capabilities
3 Execute the getMyFiles routine returning at most 100 entries.
4 If there are no values, you do not have any files
5 Create an ordered list in HTML
6 For each valid file, create a child entry to contain the ordered item
7 The display name is the first 25 characters of a file name with a '…' suffix if it is long
8 Build up an HTML tag so if a user clicks on the file they are prompted to download it
9 Provide a place to show an error message

Table 2: Notes to describe the snippet's source code

Loading the Moodle Block code into your environment

Copy and Configure the Moodle Block Code

Extract the contents of the MoodleBlocks.zip file and copy the **icfiles** directory into **$MOODLE_HOME/blocks**. As a Moodle administrator you will see the following:



Plugins check

This page displays plugins that may require your attention during the upgrade. Highlighted items include new plugins that are about to be installed, updated plugins that are about to be upgraded and any missing plugins. Add-ons are highlighted if there is an available update for them. It is recommended that you check whether there are more recent versions of add-ons available and update their source code before continuing with this Moodle upgrade.

Check for available updates

Number of plugins requiring your attention: 1

Display the full list of installed plugins

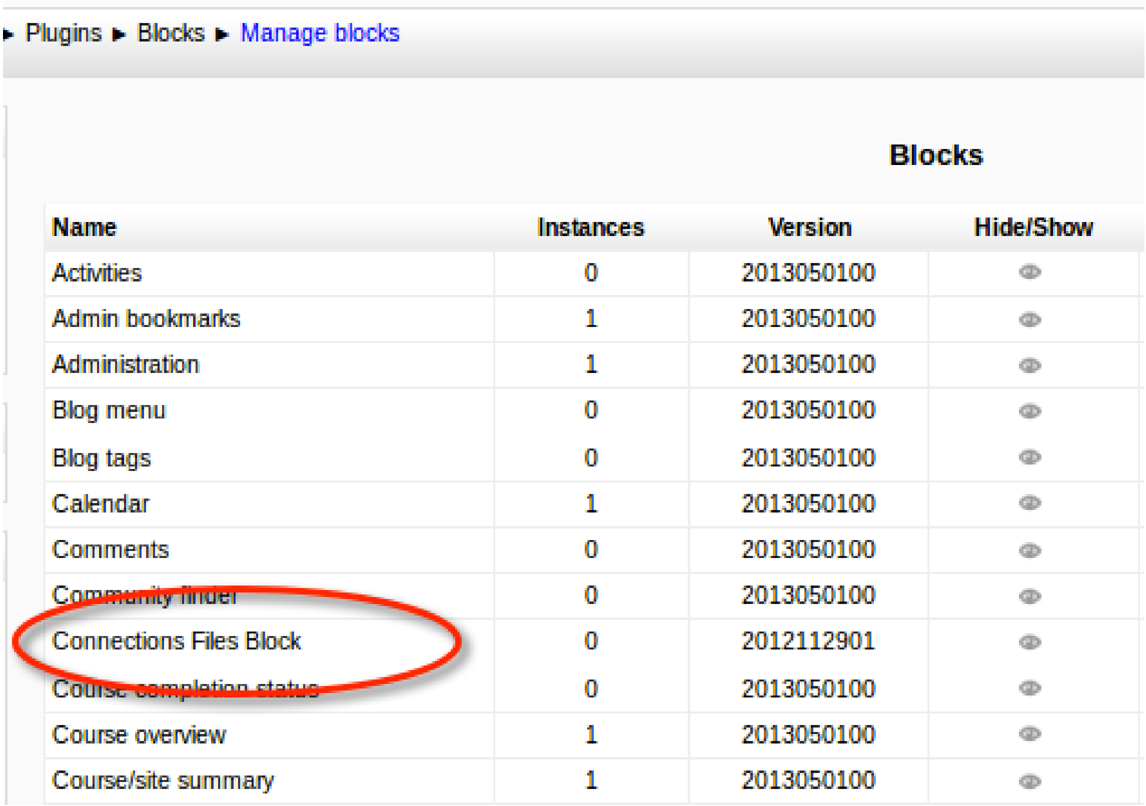
Plugin name	Directory	Source	Current version	New version	Requires	Status
			Blocks			
Connections Files Block	/blocks/icfiles	Add-on		2012112901	Moodle 2012112901	To be installed

Reload

Upgrade Moodle database now

Figure 13: Moodle is now aware of a new Block

Click on the **Upgrade Moodle database now** button. If you now go to **Site Administration** -> **Plugins-> Blocks** -> **Manage Blocks**, you will see a new Connections Files Block available.



▸ Plugins ▸ Blocks ▸ Manage blocks

Blocks

Name	Instances	Version	Hide/Show
Activities	0	2013050100	👁
Admin bookmarks	1	2013050100	👁
Administration	1	2013050100	👁
Blog menu	0	2013050100	👁
Blog tags	0	2013050100	👁
Calendar	1	2013050100	👁
Comments	0	2013050100	👁
Community finder	0	2013050100	👁
Connections Files Block	0	2012112901	👁
Course completion status	0	2013050100	👁
Course overview	1	2013050100	👁
Course/site summary	1	2013050100	👁

Figure 14: And the new Block is now available to be used

To make the block accessible in your environment, as the administrator go into **Front page settings** -> **Turn editing on**. In the lower left hand side of the screen, you will see a block containing a drop down titled **Add a block**. It provides a list of available blocks, and the **Connections Files Block** is one of them (Figure). Add it, and you will probably see a Certificate security error (Figure).

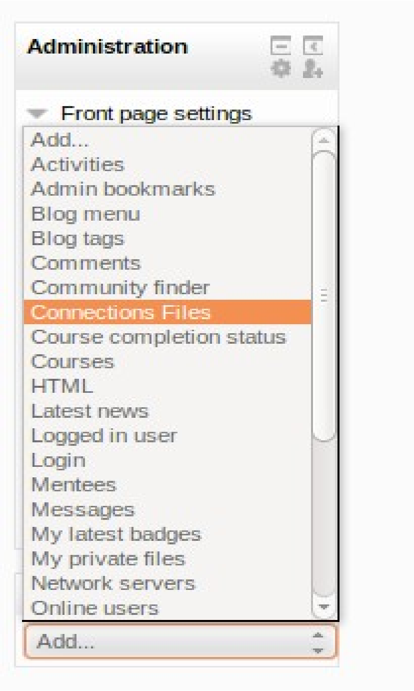


Figure 15: Making your new block visible

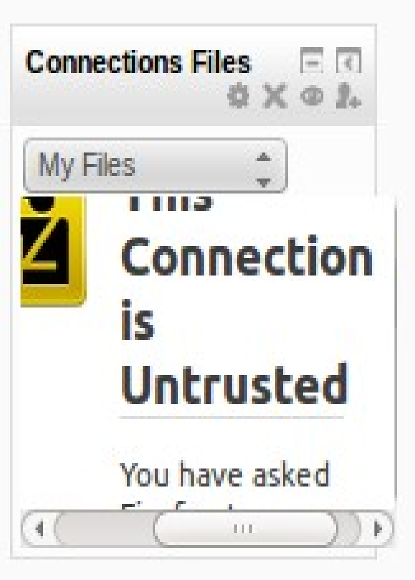
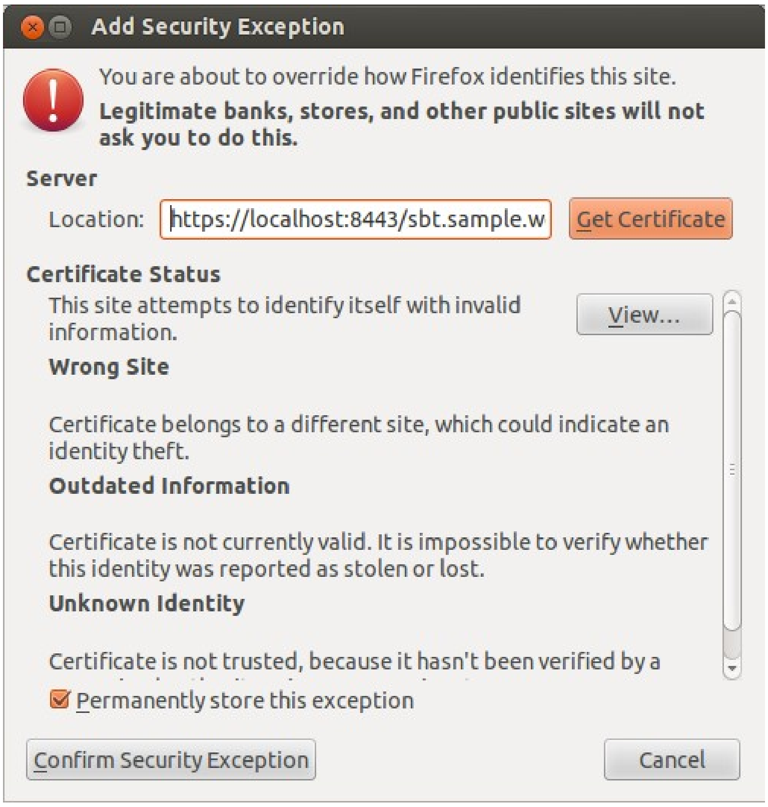


Figure 16: Your Tomcat host is untrusted

Right click within the block and open that frame in a new Tab or browser window. You will then see the whole block and can provide an exception to the security warning.



Add Security Exception

You are about to override how Firefox identifies this site.

Legitimate banks, stores, and other public sites will not ask you to do this.

Server

Location: https://localhost:8443/sbt.sample.w

Certificate Status

This site attempts to identify itself with invalid information.

Wrong Site

Certificate belongs to a different site, which could indicate an identity theft.

Outdated Information

Certificate is not currently valid. It is impossible to verify whether this identity was reported as stolen or lost.

Unknown Identity

Certificate is not trusted, because it hasn't been verified by a

☑ Permanently store this exception

Figure 17: Adding a security exception

You will then see the Connections Files block half hidden by the block format. Enter your login credentials and click on the **Log in** button. In the block (Figure) you can now see an ordered list of My Files.

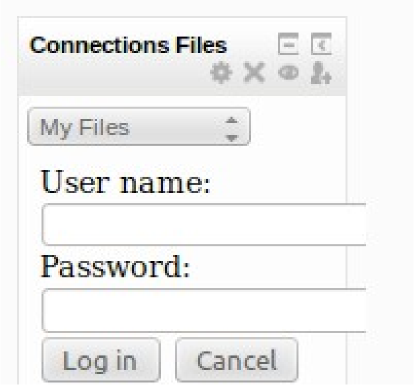
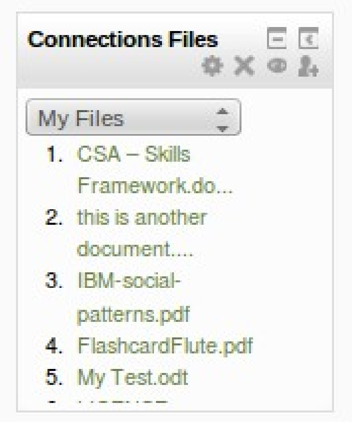


Figure 18: You now have a login prompt



1. CSA – Skills Framework.do...
2. this is another document....
3. IBM-social-patterns.pdf
4. FlashcardFlute.pdf
5. My Test.odt

Figure 19: Once logged in, files are visible

You can now add additional blocks for Communities, since this source is also available in the code from this article. You will now customise your experience.

Customising your blocks and Moodle

In Figure 19, you can see that the file list is returned in a green font. Moodle is responsible for its own theming, but it has no control over the SBTK snippet experience. Remember that the content is returned within an iFrame and is therefore not really running within Moodle, but remotely within Tomcat. This is however, easy to resolve. Each snippet is written in Javascript, but you can also provide a corresponding HTML file. This file can be used for example, to build a form or table to contain content, but I have used it to embed some CSS theming. Because my default theme is the Moodle **Boxxie** theme, I borrowed values from its CSS files. Now my Moodle theme and the Connections content look and feel the same.

```
<style type="text/css" media=screen>
        a:link,
        a:visited { color: #69804e; text-decoration: none; }
        ol { margin:0px 0; padding:0; }
        ol li { margin: 30px; padding: 0; margin-bottom: 0; margin-top: 0; }
</style>
```

Listing 5: The associated HTML file can also be used to host CSS styling

Here is a screen shot of my (empty) Moodle environment containing properly themed Files and Communities blocks.

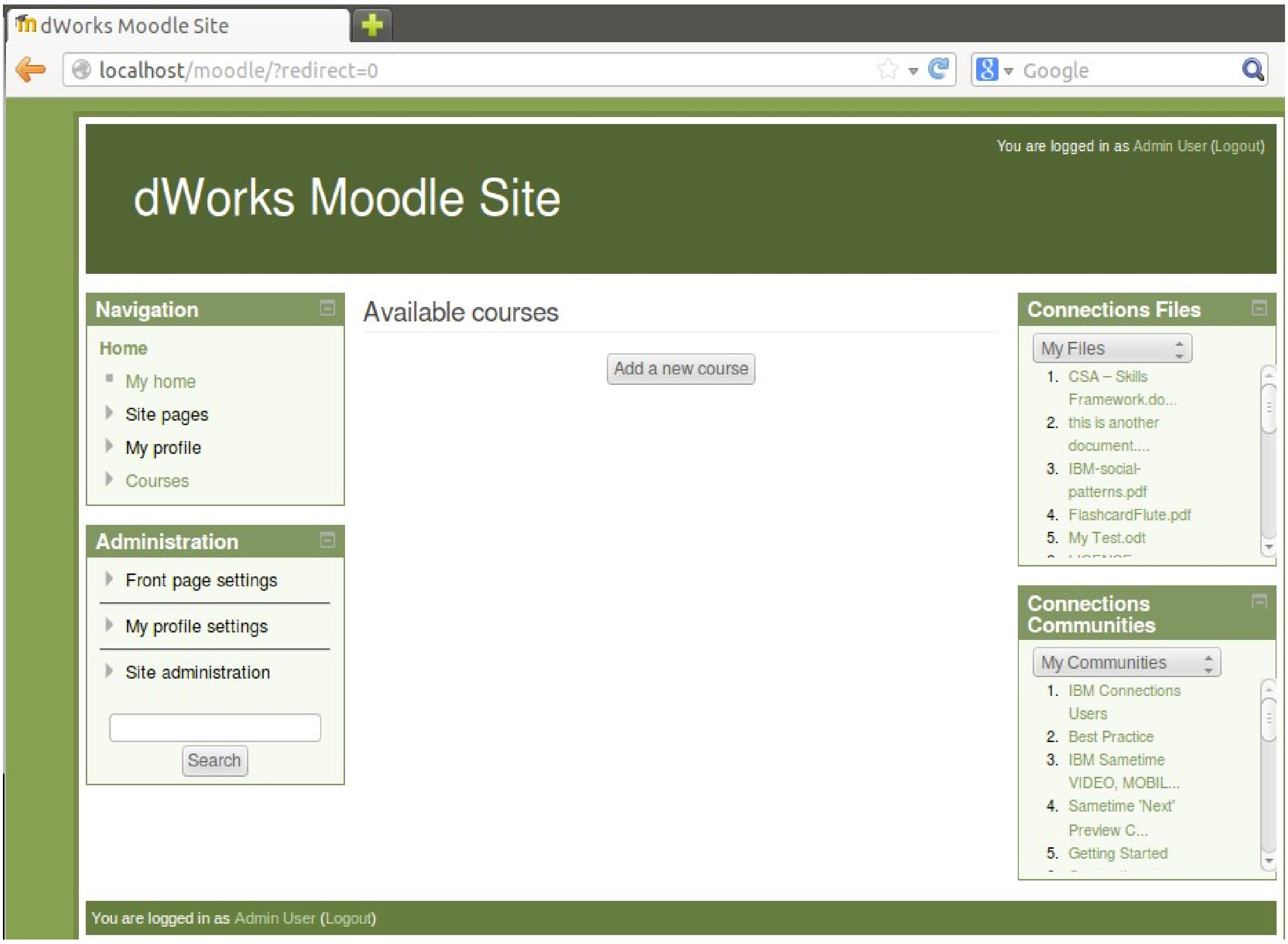


Figure 20: Themed blocks seamlessly blend into your Moodle experience.

Adding Moodle Profile Fields

Every Moodle user has a profile, which allows them to provide information such as full name, Skype ID, Mobile number, and photo. As an administrator, you can also include additional profile fields; this gives you much flexibility around how you exploit the capabilities in SmartCloud or Connections. For example, I have yet another SmartCloud Block where users can open SmartCloud meetings, or join regular class webinars. I have set these values up as Profile fields. These are user specific, so a lecturer or student can host his or her own meeting without having to remember their value, or even go to the SmartCloud home page.

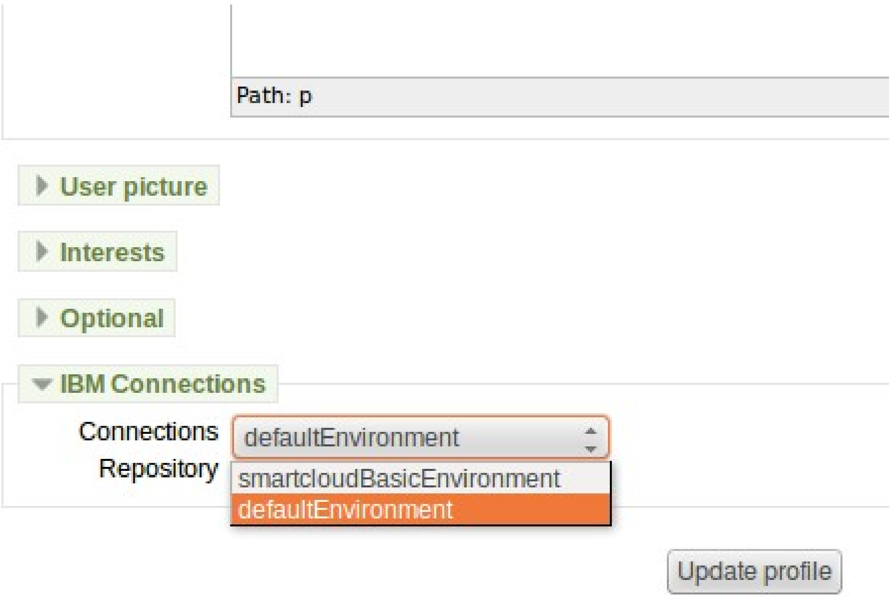
Here, I use a Profile Field to set the default Connections repository. In this case, we are using Greenhouse, or SmartCloud for Social Business.

Guidance on how to add user-defined fields can be found here: http://docs.moodle.org/20/en/User_profile_fields. In summary however, you add user-defined fields as the administrator by going to ***Settings*** > ***Site Administration*** > ***Users*** > ***Accounts*** > ***User Profile Fields*** > ***Create new Profile Field***.

First create a new Profile Category called **IBM Connections**.
Then create a new Profile Field of type **Menu of Choices**.

The short name will be **IBMICSC**
The Name will be **Connections Repository**
Set your category to be the **IBM Connections** category you just created
Your menu options will be:
smartcloudBasicEnvironment, and
defaultEnvironment
Set your default environment to be **defaultEnvironment**

Go into **My profile settings** ü **Edit profile** where you will see an option to define your Connections point as either Connections (on-premises) or Connections (SmartCloud). The variable **IBMICSC** is referenced in the block code shown in Listing 3 (note 3 from able).



Connections Repository: defaultEnvironment
smartcloudBasicEnvironment
defaultEnvironment

Figure 21: You can now choose your default Connections environment

The configuration required to access this has already been included in **$SBTK_HOME/tomcat/apache-tomcat-7.0.30/webapps/sbt.sample.web/WEB-INF/managed-beans.xml**. We will now examine the default managed-bean.

```
<!-- Default Environment -->
<managed-bean>
       <managed-bean-name>defaultEnvironment</managed-bean-name>
       <managed-bean-class>com.ibm.sbt.jslibrary.SBTEnvironment</managed-bean-class>
       <managed-bean-scope>application</managed-bean-scope>
       <managed-property>
       <property-name>endpoints</property-name>

<value>connections,connectionsOA2,smartcloud,smartcloudOA2,domino,acmeAir,conAcmeApp,connectionsSSO,dropbox,twitter</value>
       </managed-property>
        <managed-property>
       <property-name>properties</property-name>
              <value>sample.email1,sample.email2</value>
       </managed-property>
```

```
</managed-bean>
```

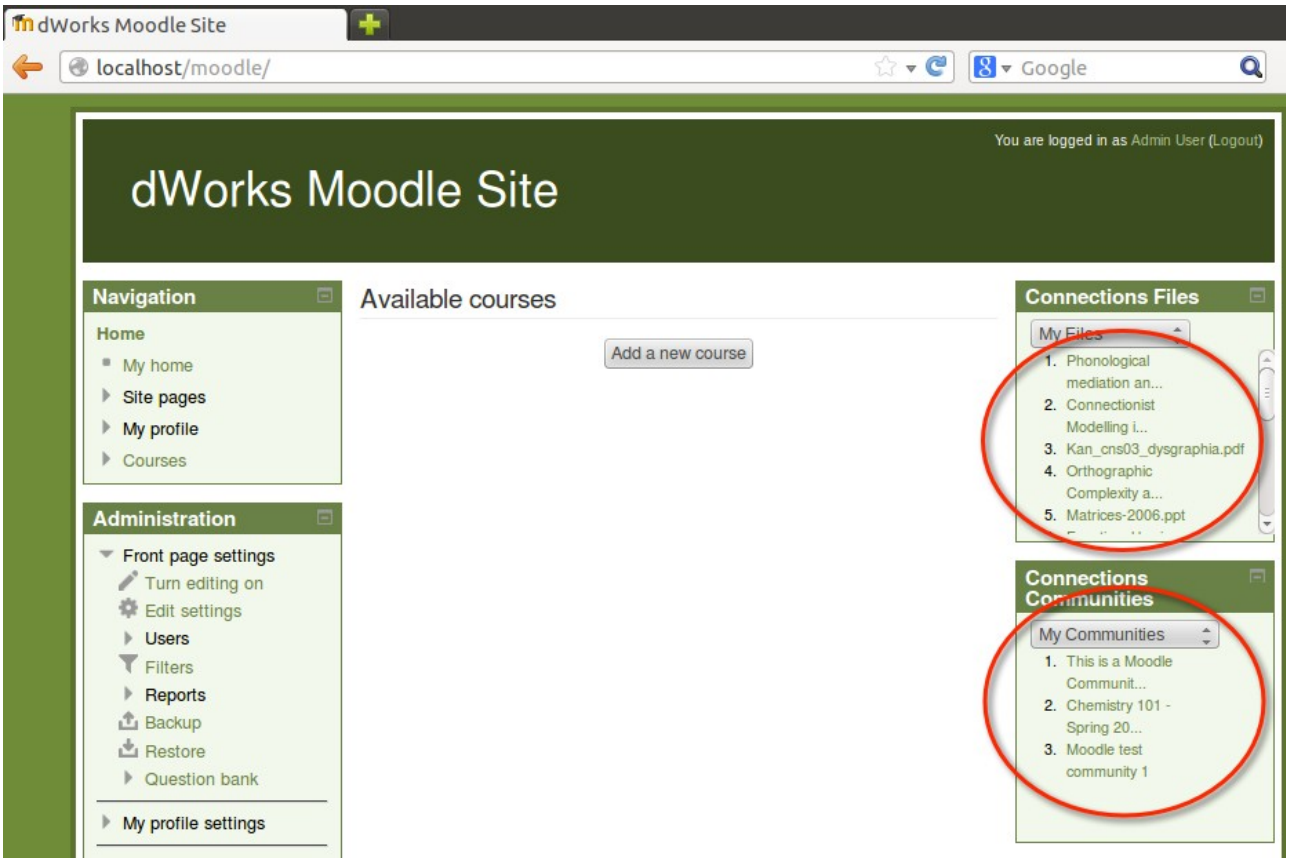
Listing 6: The default managed-bean

This is your default Basic authentication environment to an on-premises Connections site. There is also another environment set up to allow Basic authentication to SmartCloud. Note that this is different from the OAuth and OAuth2 approaches you might want to consider. Here you’re using basic authentication with a user name and password.

```
<!-- SmartCloud Basic Authentication Environment -->
<managed-bean>
      <managed-bean-name>smartcloudBasicEnvironment</managed-bean-name>
      <managed-bean-class>com.ibm.sbt.jslibrary.SBTEnvironment</managed-bean-class>
      <managed-bean-scope>application</managed-bean-scope>
      <managed-property>
            <property-name>endpoints</property-name>
            <value>smartcloudBasic:connections</value>
      </managed-property>
</managed-bean>
```

Listing 7: The SmartCloud basic environment allowing basic authentication.

To use the smartCloudBasicEnvironment, append **&env=smartcloudBasicEnvironment** to URLs calling your snippets. You can also see this in Listing 3 (note 6 from able 1). where the **$ICSCEnv** variable is referenced. Now when you change your Connections Repository to SmartCloud, you will be prompted again for a login, but the content will come from your SmartCloud account (Figure).



Connections Files

1. Phonological mediation an...
2. Connectionist Modelling i...
3. Kan_cns03_dysgraphia.pdf
4. Orthographic Complexity a...
5. Matrices-2006.ppt

Connections Communities

1. This is a Moodle Communit...
2. Chemistry 101 - Spring 20...
3. Moodle test community 1

Figure 22: My Moodle site now returns my SmartCloud files and communities.

You are not just restricted to using a single repository since you can use multiple blocks at the same time. Just copy an existing block and modify the name. Then you can show information from multiple Connections repositories simultaneously.

Conclusion

People are creatures of habit, and while Connections and SmartCloud for Social Business offer excellent capabilities, embedding these capabilities into existing day-to-day activities is key to helping them be used. In this article, I showed how simple it is to add these capabilities from multiple repositories and theme them to fit into your existing environment. In principle, users of Moodle need not be aware that the content is not part of an existing University infrastructure.

References and Links

Moodle description on Wikipedia:
https://en.wikipedia.org/wiki/Moodle

Get the Social Business Toolkit:
http://ibmsbt.openntf.org/

A step by step guide to creating a Moodle Block:

http://docs.moodle.org/dev/Blocks

Social Business Toolkit Documentation:
https://www.ibmdw.net/social/documentation/

Social Business Toolkit Playground:
https://greenhouse.lotus.com/sbt/SBTPlayground.nsf/Home.xsp

Register for a Greenhouse account:
http://greenhouse.lotus.com and then choose ‘Sign Up’ in the top right corner.

Register for a Trial SmartCloud for Social Business Account:
https://www.ibm.com/cloud-computing/social/us/en/startatrial/

Download the code for the blocks and snippets used in this article here:
http://www-10.lotus.com/ldd/lcwiki.nsf/dx/Extending_Moodle_with_IBM_Social_Business_Toolkit