Icodeon SCORM Player Integration

About the Integration

This integration adds a new content type into Resources. This allows you to upload a SCORM package into Resources and registers the content package with Icodeon. Clicking on the name of the resource will launch the Icodeon SCORM player. The advantage to this sort of integration is that SCORM can be leveraged anywhere in Sakai that integrates with Resources via the file picker.

Icodeon posts statistics back into tables that exist in the Sakai database. At this point there are no reports or other features that take advantage of this data. But since we pass the site and learner information to Icodeon, its conceivable this data can be queried for any number of use cases. We'd love for others to get involved to help build out the functionality in this area.

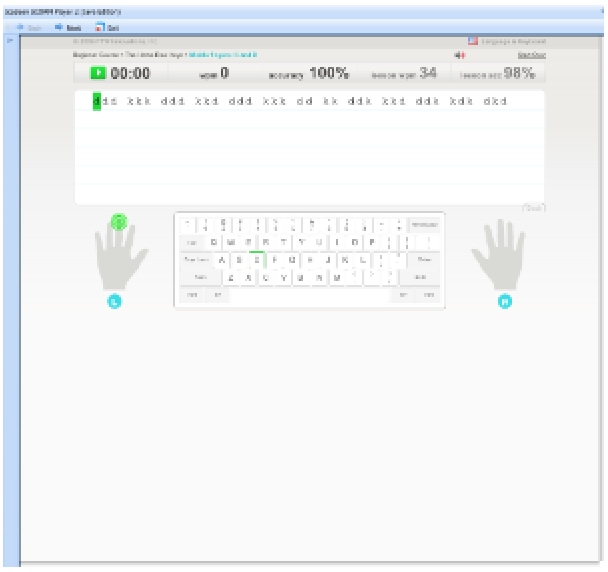
The current integration works with version 2.0.5 of Icodeon.

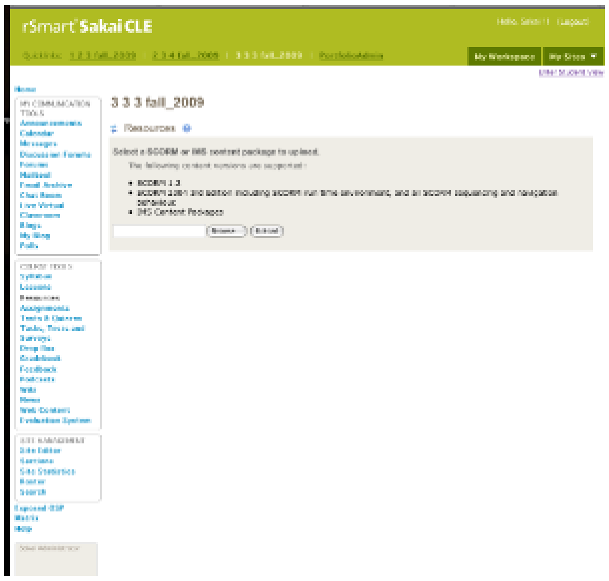
Screencast Demo

YouTube Demo

Screencast.com Demo

Screenshots

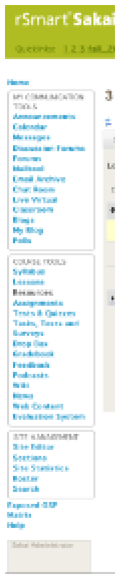




Select a SCORM or IMS content package to upload.

The following content versions are supported:

- SCORM 1.2
- SCORM 2004 3rd Edition including SCORM run time environment, and all SCORM sequencing and navigation behavior
- IMS Content Packages



Installation

The version is contrib works against a 2.6 Sakai code base, if you want to run on 2.5 you may need to back port the pom.xml's.

You'll need to grab the Icodeon components for Sakai, the rSmart common contrib module, as well as modify the Icodeon war package.

rSmart common library

You'll want to check this out into an existing Sakai working directory, in the standard contrib way.

2.6.x version

```
svn co https://source.sakaiproject.org/contrib/rsmart/rsmart-common/trunk/
 rsmart-common
```

2.5.x version

```
svn co https://source.sakaiproject.org/contrib/rsmart/rsmart-common/branches/2.5.x rsmart-common
```

To build the rsmart-common components add the rsmart-common module the top level pom.xml for Sakai, or manually build from within the rsmart-common module, like this:

```
mvn install sakai:deploy
```

Sakai Icodeon Components

The Icodeon components for Sakai can be found in the following module:

```
svn co  https://source.sakaiproject.org/contrib/rsmart/icodeon icodeon
```

You'll want to check this out into an existing Sakai working directory, in the standard contrib way.

To build the Sakai Icodeon components, add the icodeon module the top level pom.xml for Sakai, or manually build from within the icodeon module, like this:

```
mvn install sakai:deploy
```

component/

This module contains the content resource type component

impl/

This module contains the content resource type implementation

tool/

This module contains the content resource type associated front end helpers that are necessary to load SCORM packages into Sakai.

The Icodeon launching url is managed in IcodeonHttpAccess. It should be easy to use this existing integration with another player by simply creating an alternative implementation for this class.

icodeon/

The Icodeon war needs to be modified to apply a custom "PlugIn_Resolver", and adjust Icodeon to leverage the same hibernate session as Sakai. This module contains the necessary code and build script to rebuild and deploy the Icodeon War

Steps to Rebuild the Icodeon WAR

This part is not built into the normal Sakai maven build, you'll need to perform the following additional steps in order to deploy Icodeon into Sakai's tomcat:

1. download and extract the icodeon distribution
2. modify the paths in the icodeon/build.sh script. The names should be self documenting

```
MAVEN_CMD=mvn-trunk
TOMCAT_HOME=/Users/jbush/Dev/tools/apache-tomcat-5.5.20-trunk/
ICODEON_DIR=/Users/jbush/Dev/projects/sakai-2.6.0/icodeon/player-2.0.5/
LICENSE_FILE=/Users/jbush/Desktop/player2.license
```

3. cd icodeon
4. run the build script

   ```
   ./build.sh
   ```

5. restart tomcat

   ⚠ You'll need to convert the build script to bat or ant if you aren't in a unix friendly environment.

Here's what the build script does:

```
#!/bin/sh

MAVEN_CMD=mvn-trunk
TOMCAT_HOME=/Users/jbush/Dev/tools/apache-tomcat-5.5.20-trunk/
ICODEON_DIR=/Users/jbush/Dev/projects/sakai-2.6.0/icodeon/player-2.0.5/
SRC_DIR=src
LICENSE_FILE=/Users/jbush/Desktop/player2.license

CURRENT_PATH=`pwd`

cd $SRC_DIR
# install the icodeon jar needed as a dependency
$MAVEN_CMD install:install-file -DgroupId=com.icodeon.player -DartifactId=icodeon-player -Dversion=2 -
Dpackaging=jar -Dfile=$ICODEON_DIR/webapps/player2/WE
B-INF/lib/icodeon-player-2.jar
# build the integration code
$MAVEN_CMD clean install sakai:deploy
cp icodeon-player-plugins/target/icodeon-player-plugins-2.jar $ICODEON_DIR/webapps/player2/WEB-INF/lib
# copy the jars to the staging webapp
cp vendor-player-plugins/target/vendor-player-plugins-2.jar $ICODEON_DIR/webapps/player2/WEB-INF/lib
cp $LICENSE_FILE  $ICODEON_DIR/webapps/player2/WEB-INF/classes
# copy our web.xml to the staging area
cp ../webapps/player2/WEB-INF/web.xml $ICODEON_DIR/webapps/player2/WEB-INF/
# remove hibernate and ehcache, these are in tomcat/shared
rm -f $ICODEON_DIR/webapps/player2/WEB-INF/lib/hibernate*
rm -f $ICODEON_DIR/webapps/player2/WEB-INF/lib/ehcache*
# deploy the webapp
cp -R $ICODEON_DIR/webapps/player2 $TOMCAT_HOME/webapps
cd $CURRENT_PATH
```

Configuration

To enable the scorm content resource type you need to add the following to your sakai or local.properties

```
enable.scorm=true
```

Technical Information about the Integration

The SakaiCHSResolver is the Sakai custom PlugIn_resolver that is responsible for turning the course_id parameter passed to the player into content. The course_id is actually a reference to content stored in CHS. The resolver extracts the content onto the file system and the player takes it from there.

There are modifications to the provided HibernateUtil.java file to enable a shared hibernate session between Icodeon and Sakai. The work primarily involved moving the existing Icodeon integration code they ship with out of the /maven folder and a shared library containing the model classes, and a component that registers the hbms. Because of Sakai's component manager and handling of shared libraries, this is somewhat a manual process. I wish there was a better way to handle this moving forward, its making deploying Icodeon updates a little tricky. I tried the normal vendor drop approach here, without much luck, because things moved around too much.

There are some modifications to the web.xml to reference the custom plugin and register the persistence engine that captures the learner activity.