

# Web Contents loading via Batch Process

Zaninello Simone, Technical Leader, Ariadne

# Introduzione 1/2

Quando abbiamo la necessità di importare dei contenuti?

- Durante lo sviluppo (in particolare di tema e modelli) per avere un riscontro immediato.
- Dopo una migrazione abbiamo un pregresso da riportare.
- Dobbiamo alimentare periodicamente Liferay da fonti esterne.

# Introduzione 2/2

## Cosa Liferay ci offre come tool di importazione?

- Resource Importer  
... è un tool che permette a chi sviluppa un tema di includere files e contenuti che vengano importati automaticamente nel portale nel momento in cui il tema viene rilasciato.  
Le nuove risorse possono essere importate in un “Site” o in un “Site Template”.

<https://www.liferay.com/it/documentation/liferay-portal/6.2/development/-/ai/importing-resources-with-your-themes-liferay-portal-6-2-dev-guide-09-en>

# Il nostro caso d'uso

L'importazione massiva di Web Content tramite un processo batch

- Creare/modificare automaticamente contenuti da una sorgente dati esterna
- Usare il CMS per apportare modifiche o gestire il ciclo di vita e pubblicazione dei contenuti importati

Perché no in resource importer? Perché quest'ultimo è pensato per un caricamento una tantum iniziale.

# Come funziona? 1/6

Configurazione preliminare del portale: prevedere le seguenti properties in fase di configurazione (portal-ext.properties):

```
# Abilitare la possibilità di definire manualmente le key
# di strutture e template
dynamic.data.mapping.structure.force.autogenerate.key=false
dynamic.data.mapping.template.force.autogenerate.key=false

# solo perché nell'esempio si usa l'italiano
locales.enabled=\
    ca_ES,zh_CN,en_US,fi_FI,fr_FR,de_DE,iw_IL,hu_HU,ja_JP,pt_BR,es_ES,it_IT
```

# Come funziona? 2/6 (sorgente JSON)

```
1 [
2   {
3     "name": "BOOTCAMP",
4     "contents": [
5       {
6         "id": "1",
7         "en_US": {
8           "title": "Web content sync",
9           "speaker": "Zaninello Simone",
10          "description": "Web content sync"
11        },
12        "it_IT": {
13          "title": "Sincronizzazione Wen Content Modificato",
14          "speaker": "Zaninello Simone",
15          "description": "Sincronizzazione Wen Content Modificato"
16        }
17      },
18      {
19        "id": "2",
20        "en_US": {
21          "title": "Web content sync 2",
22          "speaker": "Zaninello Simone",
23          "description": "Web content sync 2"
24        },
25        "it_IT": {
26          "title": "Sincronizzazione Wen Content 2",
27          "speaker": "Zaninello Simone",
28          "description": "Sincronizzazione Wen Content 2"
29        }
30      }
31    ]
32  }
33 ]
```

Rimappa la la chiave  
struttura creata in Liferay

Uno per ogni lingua del  
sito

Ogni campo rimappa  
quello definito nella  
struttura

# Come funziona? 3/6

Creare le strutture sulla base delle entità che si dovranno importare.


## Structures

View [Source](#)

Default Language:

 English (United States) [Change](#) [Add Translation](#)

Available Translations:

 Italian (Italy) [x](#)

[Fields](#) [Settings](#)

Property Name	Value
Type	text
Field Label	Title
Show Label	Yes
Required	No
Name	title
Predefined Value	
Tip	

Title

Speaker

Description

## Structures

Name (Required)

Bootcamp



[Details](#)

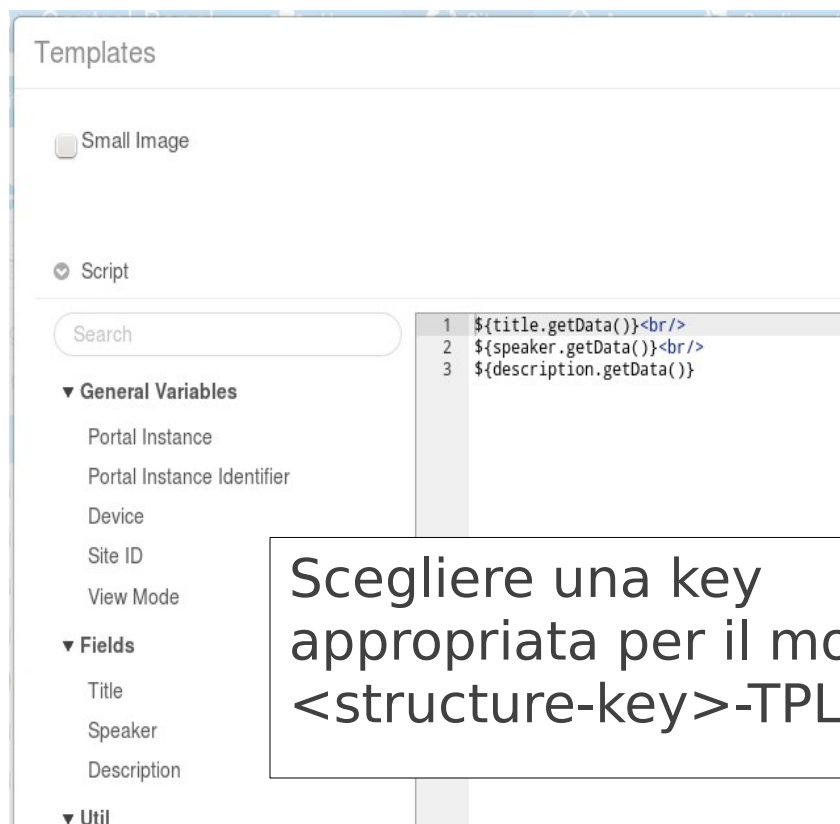
Structure Key

BOOTCAMP

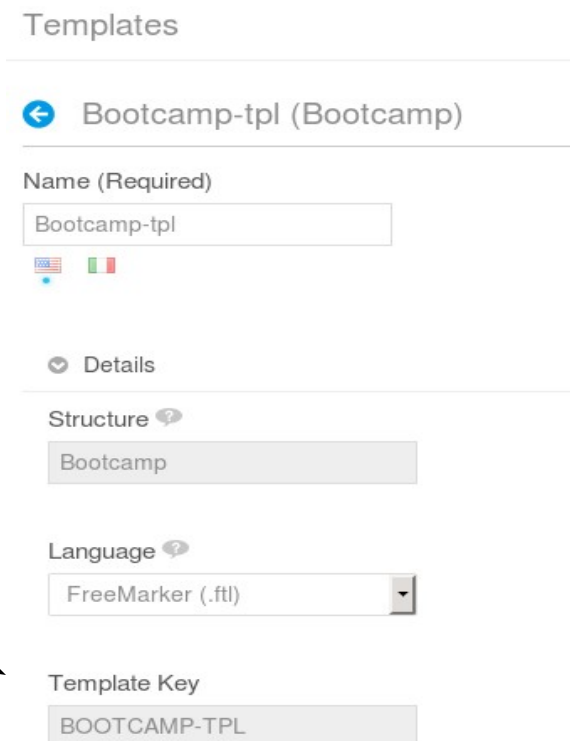
Scegliere una key appropriata per la struttura

# Come funziona? 4/6

Creare i modelli sulla base delle strutture individuate.



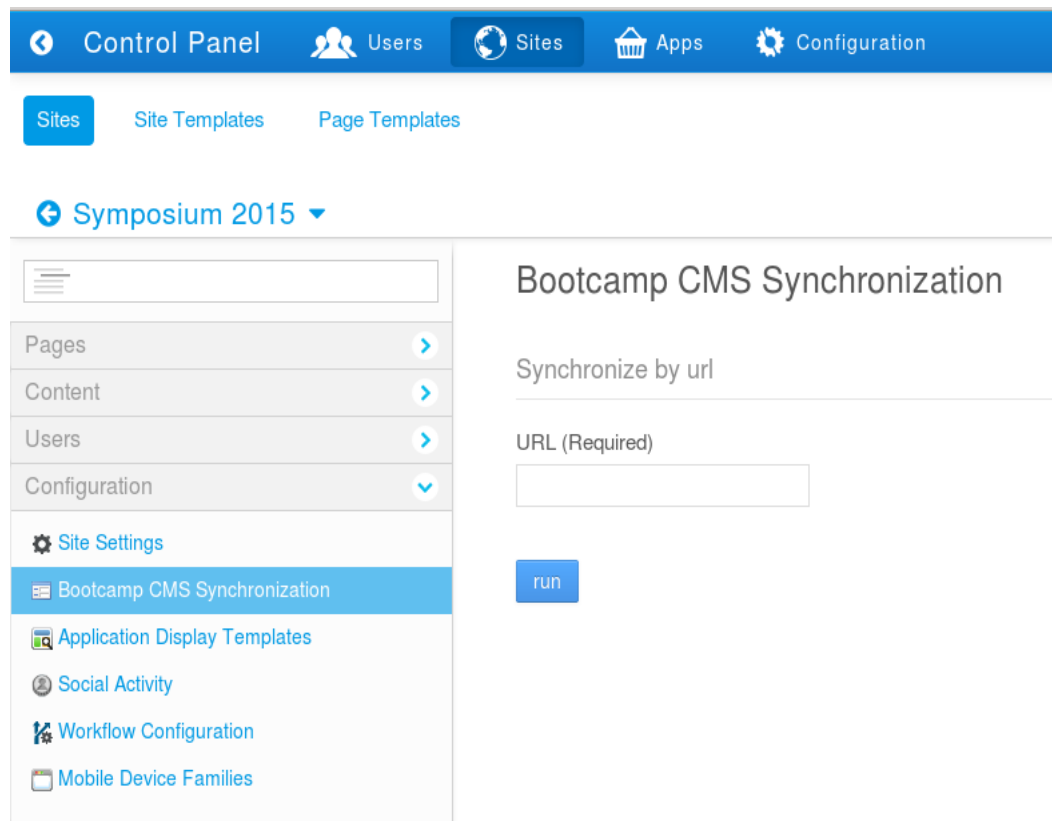
Scegliere una key appropriata per il modello:  
<structure-key>-TPL





# Come funziona? 5/6

Attivazione del processo tramite una portlet presente nella sezione di configurazione sito: l'importazione avviene sul sito da cui viene attivato il processo.

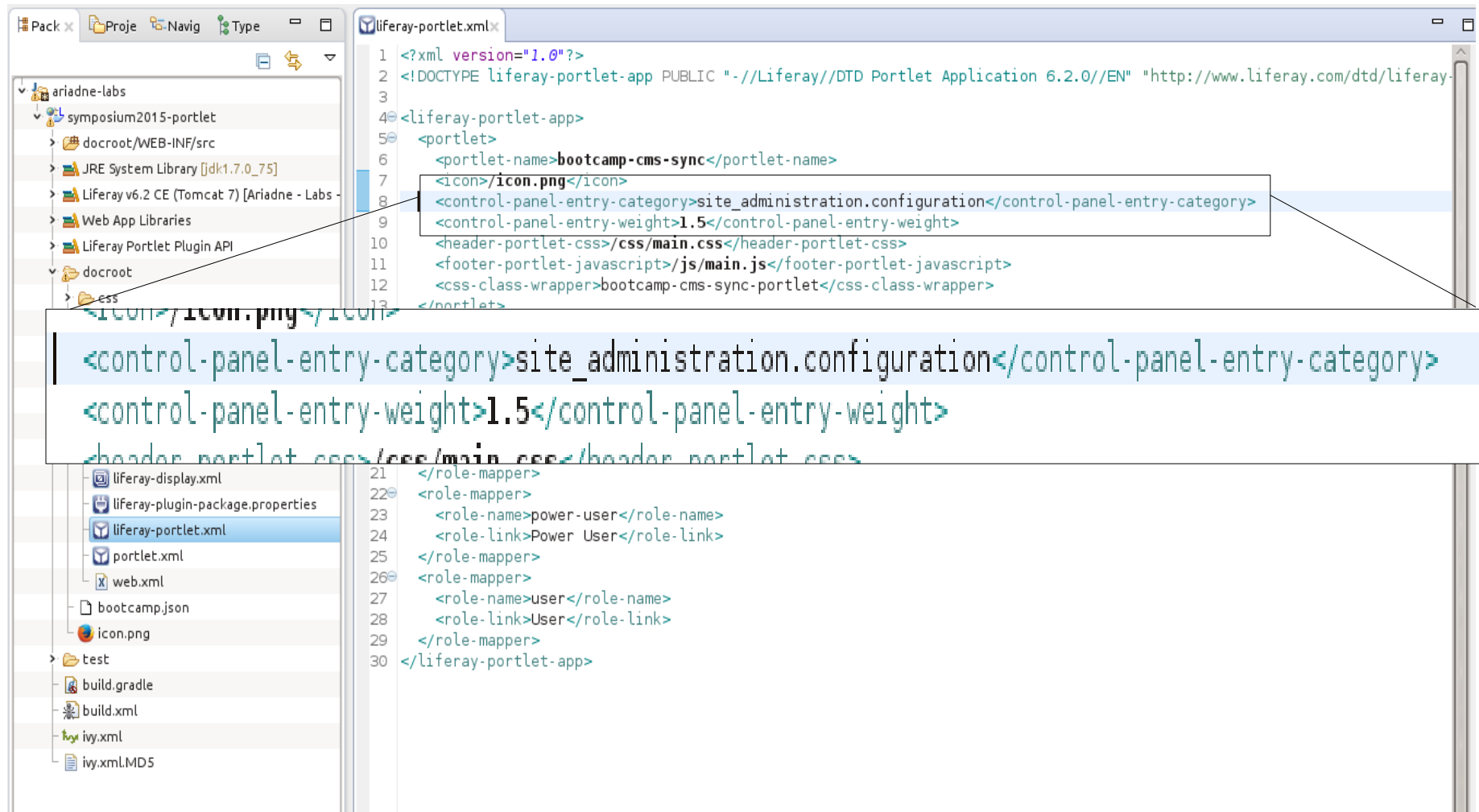


# Come funziona? 6/6

Il processo sfrutta i seguenti framework di Liferay:

- Background Task: attivazione del processo
- JSON Util: lettura dei dati presenti nella sorgente JSON
- Dynamic Data Mapping API: conversione del dato nel formato Liferay
- Journal Article API: storicizzazione del dato

# Definizione Portlet di configurazione sito



The screenshot shows an IDE with a project named 'ariadne-labs' and a file named 'liferay-portlet.xml'. The XML content is as follows:

```
1 <?xml version="1.0"?>
2 <!DOCTYPE liferay-portlet-app PUBLIC "-//Liferay//DTD Portlet Application 6.2.0//EN" "http://www.liferay.com/dtd/liferay-
3
4 <liferay-portlet-app>
5   <portlet>
6     <portlet-name>bootcamp-cms-sync</portlet-name>
7     <icon>/icon.png</icon>
8     <control-panel-entry-category>site_administration.configuration</control-panel-entry-category>
9     <control-panel-entry-weight>1.5</control-panel-entry-weight>
10    <header-portlet-css>/css/main.css</header-portlet-css>
11    <footer-portlet-javascript>/js/main.js</footer-portlet-javascript>
12    <css-class-wrapper>bootcamp-cms-sync-portlet</css-class-wrapper>
13  </portlet>
14
21 </role-mapper>
22 <role-mapper>
23   <role-name>power-user</role-name>
24   <role-link>Power User</role-link>
25 </role-mapper>
26 <role-mapper>
27   <role-name>user</role-name>
28   <role-link>User</role-link>
29 </role-mapper>
30 </liferay-portlet-app>
```

A callout box highlights the following XML elements:

```
<control-panel-entry-category>site_administration.configuration</control-panel-entry-category>
<control-panel-entry-weight>1.5</control-panel-entry-weight>
```

# Attivazione di un Background Task

```
BootcampCMSSyncPortlet.java
1 package it.ariadne.symposium2015.bootcamp.cmssync.portlet;
2
3 import java.io.Serializable;
4
22
23 public class BootcampCMSSyncPortlet extends MVCPortlet {
24
25     public void runSync(ActionRequest actionRequest, ActionResponse actionResponse)
26         throws PortalException, SystemException {
27
28         ThemeDisplay themeDisplay = (ThemeDisplay) actionRequest
29             .getAttribute(WebKeys.THEME_DISPLAY);
30
31         long userId = themeDisplay.getUserId();
32         long groupId = themeDisplay.getScopeGroupId();
33
34         String url = ParamUtil.getString(actionRequest, "url");
35
36         ServiceContext serviceContext = ServiceContextFactory
37             .getInstance(actionRequest);
38         serviceContext.setRequest(null);
39
40         HashMap<String, Serializable> taskContextMap = new HashMap<String, Serializable>();
41         taskContextMap.put("userId", userId);
42         taskContextMap.put("groupId", groupId);
43         taskContextMap.put("url", url);
44         taskContextMap.put("serviceContext", serviceContext);
45
46         if (!BootcampCMSSyncUtil.isSynchronizationRunning(groupId)) {
47             BackgroundTaskLocalServiceUtil.addBackgroundTask(userId, groupId,
48                 "BOOTCAMP_CMS_SYNC",
49                 ServletContextPool.keySet().toArray(new String[0]),
50                 BootcampCMSSyncTaskExecutor.class, taskContextMap,
51                 new ServiceContext());
52         }
53     }
54 }
55 }
```

# Esecuzione di un Background Task

```
BootcampCMSSyncTaskExecutor.java
1 package it.ariadne.symposium2015.bootcamp.cmssync.job;
2
3 import java.io.Serializable;
4
5
6
7
8
9
10
11
12
13
14
15
16
17 public class BootcampCMSSyncTaskExecutor extends BaseBackgroundTaskExecutor {
18
19     private static Log _log = LogFactoryUtil
20         .getLog(BootcampCMSSyncTaskExecutor.class);
21
22     @Override
23     public BackgroundTaskResult execute(BackgroundTask task) throws Exception {
24
25         Map<String, Serializable> taskContextMap = task.getTaskContextMap();
26
27         long userId = MapUtil.getLong(taskContextMap, "userId");
28         long groupId = MapUtil.getLong(taskContextMap, "groupId");
29         String url = MapUtil.getString(taskContextMap, "url");
30         ServiceContext serviceContext = (ServiceContext) taskContextMap
31             .get("serviceContext");
32
33         BackgroundTaskResult result = null;
34         try {
35             String taskReport = BootcampCMSSyncUtil.synchronize(userId, groupId, url,
36                 serviceContext);
37
38             result = new BackgroundTaskResult(
39                 BackgroundTaskConstants.STATUS_SUCCESSFUL, taskReport);
40         } catch (Throwable e) {
41             if (_log.isDebugEnabled()) {
42                 _log.error(e, e);
43             }
44
45             result = new BackgroundTaskResult(BackgroundTaskConstants.STATUS_FAILED,
46                 e.getMessage());
47         }
48
49         return result;
50     }
51 }
```

# Verifica di un Background Task in esecuzione

```
public static boolean isSynchronizationRunning(long groupId) {
    try {
        DynamicQuery dynamicQuery = DynamicQueryFactoryUtil
            .forClass(BackgroundTask.class);

        Criterion criterion = RestrictionsFactoryUtil.eq("status", new Integer(
            BackgroundTaskConstants.STATUS_NEW));
        criterion = RestrictionsFactoryUtil.or(criterion, RestrictionsFactoryUtil
            .eq("status", new Integer(BackgroundTaskConstants.STATUS_QUEUED)));
        criterion = RestrictionsFactoryUtil.or(criterion,
            RestrictionsFactoryUtil.eq("status", new Integer(
                BackgroundTaskConstants.STATUS_IN_PROGRESS)));
        criterion = RestrictionsFactoryUtil.and(criterion,
            RestrictionsFactoryUtil.eq("groupId", groupId));
        criterion = RestrictionsFactoryUtil.and(criterion,
            RestrictionsFactoryUtil.eq("taskExecutorClassName",
                BootcampCMSSyncTaskExecutor.class.getName()));

        dynamicQuery.add(criterion);

        return BackgroundTaskLocalServiceUtil.dynamicQueryCount(dynamicQuery) > 0;
    } catch (Throwable e) {
        if (_log.isErrorEnabled()) {
            _log.error(e, e);
        }
        return false;
    }
}
```

# Caricamento sorgente dati JSON 1/2

```
1 [
2   {
3     "name": "BOOTCAMP",
4     "contents": [
5       {
6         "id": "1",
7         "en_US": {
8           "title": "Web content sync",
9           "speaker": "Zaninello Simone",
10          "description": "Web content sync"
11        },
12        "it_IT": {
13          "title": "Sincronizzazione Wen Content Modificato",
14          "speaker": "Zaninello Simone",
15          "description": "Sincronizzazione Wen Content Modificato"
16        }
17      },
18      {
19        "id": "2",
20        "en_US": {
21          "title": "Web content sync 2",
22          "speaker": "Zaninello Simone",
23          "description": "Web content sync 2"
24        },
25        "it_IT": {
26          "title": "Sincronizzazione Wen Content 2",
27          "speaker": "Zaninello Simone",
28          "description": "Sincronizzazione Wen Content 2"
29        }
30      }
31    ]
32  }
33 ]
```

Rimappa la la chiave  
struttura creata in Liferay

Uno per ogni lingua del  
sito

Ogni campo rimappa  
quello definito nella  
struttura

# Caricamento sorgente dati JSON 2/2

```
private static JSONArray getJSONContents(String url) throws JSONException,
    IOException {
    Options options = new Options();
    options.setLocation(url);

    return JSONFactoryUtil.createJSONArray(HttpUtil.URLToString(options));
}
```

- Si fa uso del framework Http per il recupero della sorgente dati JSON.



# Inserimento/Modifica web content 1/3

```
for (int i = 0; i < jsonStructures.length(); i++) {  
    JSONObject jsonStructure = jsonStructures.getJSONObject(i);  
    String structureKey = StringUtil.toUpperCase(jsonStructure  
        .getString("name"));  
  
    DDMStructure structure = DDMStructureLocalServiceUtil.fetchStructure(  
        groupId, classNameId, structureKey, false);  
  
    if (structure != null) {  
        JSONArray jsonContents = jsonStructure.getJSONArray("contents");  
  
        for (int j = 0; j < jsonContents.length(); j++) {  
  
            JSONObject jsonContent = jsonContents.getJSONObject(j);  
  
            String contentId = jsonContent.getString("id");  
            String articleId = structureKey + StringPool.UNDERLINE + contentId;
```

# Inserimento/Modifica web content 2/3

```
Fields fields = null;
```

```
for (Locale locale : siteLocales) {  
    String languageId = LocaleUtil.toLanguageId(locale);  
    JSONObject jsonLocale = jsonContent.getJSONObject(languageId);  
  
    if (jsonLocale != null) {  
        List<String> fieldsDisplayValues = new ArrayList<String>();  
        Map<String, Serializable> attributes = new HashMap<String, Serializable>();  
        attributes.put("defaultLanguageId",  
            LocaleUtil.toLanguageId(siteDefaultLocale));  
        attributes.put("toLanguageId", languageId);  
        attributes.put("languageId", languageId);  
        ...  
    }  
}
```

```
{  
    "id": "1",  
    "en_US": {  
        "title": "Web content sync",  
        "speaker": "Zaninello Simone",  
        "description": "Web content sync"  
    },  
    "it_IT": {  
        "title": "Sincronizzazione Wen Cor",  
        "speaker": "Zaninello Simone",  
        "description": "Sincronizzazione V"  
    },  
}
```

```
for (String fieldName : structure.getFieldNames()) {  
    if (!fieldName.equals("_fieldsDisplay")) {  
        String name = fieldName + PortletConstants.INSTANCE_SEPARATOR  
            + StringUtil.randomString();  
        attributes.put(name,  
            GetterUtil.getString(jsonLocale.getString(fieldName)));  
        fieldsDisplayValues.add(name);  
    }  
}  
  
attributes.put("_fieldsDisplay", StringUtil.merge(fieldsDisplayValues));  
  
serviceContext.setAttributes(attributes);  
  
Fields newFields = DDMUtil.getFields(structure.getStructureId(),  
    serviceContext);  
if (fields == null) {  
    fields = newFields;  
} else {  
    fields = DDMUtil.mergeFields(newFields, fields);  
}  
}  
  
return JournalConverterUtil.getContent(structure, fields);
```

# Inserimento/Modifica web content 3/3

## MODIFICA

```
JournalArticle article = null;
try {
    article = JournalArticleLocalServiceUtil.getLatestArticle(
        groupId, articleId);

    article = JournalArticleLocalServiceUtil.updateArticle(userId,
        groupId, article.getFolderId(), articleId,
        article.getVersion(), titleMap, article.getDescriptionMap(),
        content, article.getLayoutUuid(), serviceContext);
} catch (NoSuchArticleException nsae) {
```

```
    Calendar calendar = CalendarFactoryUtil.getCalendar();
```

```
    int displayDateMonth = calendar.get(Calendar.MONTH);
    int displayDateDay = calendar.get(Calendar.DAY_OF_MONTH);
    int displayDateYear = calendar.get(Calendar.YEAR);
    int displayDateHour = calendar.get(Calendar.HOUR_OF_DAY);
    int displayDateMinute = calendar.get(Calendar.MINUTE);
```

```
    article = JournalArticleLocalServiceUtil.addArticle(userId,
        groupId, 0, JournalArticleConstants.CLASSNAME_ID_DEFAULT, 0,
        articleId, false, 1, titleMap, null, content, "general",
        structureKey, structureKey + "-TPL", null, displayDateMonth,
        displayDateDay, displayDateYear, displayDateHour,
        displayDateMinute, 0, 0, 0, 0, 0, 0, true, 0, 0, 0, 0, 0, true,
        true, false, null, null, null, null, serviceContext);
```


```
}
```


## INSERIMENTO

# Qual è il risultato? 1/3

☐ Add ▾    Sort By ▾    Manage ▾

Home





**[Web content sync](#)**  
Version: 1.3  
Last updated: 5 Hours ago by LP Admin  
Display Date: 11/3/15 8:21 AM







**[Web content sync 2](#)**  
Version: 1.3  
Last updated: 5 Hours ago by LP Admin  
Display Date: 11/3/15 8:23 AM

```
1 [
2   {
3     "name": "BOOTCAMP",
4     "contents": [
5       {
6         "id": "1",
7         "en_US": {
8           },
9         "it_IT": {
10          }
11      },
12      {
13        "id": "2",
14        "en_US": {
15          },
16        "it_IT": {
17          }
18      }
19    ]
20  }
21 ]
```


# Qual è il risultato? 2/3

Structure:

Bootcamp  Select (Use Default)

Default Language:  **English (United States)** Change

Available Translations:

 **Italian (Italy)**

Title (Required)

Web content sync

Title

Web content sync

Speaker

Zaninello Simone

Description

Web content sync

Templat

```
{
  "name": "BOOTCAMP",
  "contents": [
    {
      "id": "1",
      "en_US": {
        "title": "Web content sync",
        "speaker": "Zaninello Simone",
        "description": "Web content sync"
      },
      "it_IT": {
        "title": "Sincronizzazione Wen Content Modificato",
        "speaker": "Zaninello Simone",
        "description": "Sincronizzazione Wen Content Modificato"
      }
    }
  ],
}
```

# Qual è il risultato? 3/3

## Web Content Translation

### Sincronizzazione Wen Content Modificato

Translating Web Content to Italian (Italy) 🇮🇹

Title (Required)

Sincronizzazione Wen Content M

Title

Sincronizzazione Wen Content M

Speaker

Zaninello Simone

Description

Sincronizzazione Wen Content  
Modificato

```
{
  "name": "BOOTCAMP",
  "contents": [
    {
      "id": "1",
      "en_US": {
        "title": "Web content sync",
        "speaker": "Zaninello Simone",
        "description": "Web content sync"
      },
      "it_IT": {
        "title": "Sincronizzazione Wen Content Modificato",
        "speaker": "Zaninello Simone",
        "description": "Sincronizzazione Wen Content Modificato"
      }
    }
  ],
}
```

# Possibili evoluzioni

- Reperimento sorgente JSON tramite upload o Document Library
- Gestione della modifica tramite confronto di date
- Gestione dello stato dei web content
- Gestione con ambiente di staging attivo