

getEntries Tag

Argument:

<i>Argument</i>	<i>Description</i>	<i>Required/Optional</i>
parentId	Id of parent topic (aka: theme)	required

Description:

`getEntries` returns a collection of Entry objects where that entry is associated with the `parentId` provided. The Entry objects are sorted by their `lastModifyDate`. Conceptually, the returned collection would be equivalent to the collection with the following SQL:

```
SELECT * FROM entry, entry_topics
WHERE entry.id = entry_topics.entry
AND entry_topics.topic = topic
ORDER BY entry.lastModDate;
```

Tag body:

The tag body will expect a collection of Entry objects to be returned in the page-scoped variable `entries`. That collection will be iterated upon in the tag body.

Example:

On the Research Topic page, the following code iterates over all the child topics for a parent topic (theme), and then all the entries contained in a parent topic (theme). We assume we have an object called `parentTopic` which is the parent topic or theme we are displaying:

```
<isse:getTopics theme="{parentTopic.id}">
  <c:forEach var="topic" items="{topics}">
    <p> <big class="c"> {topic.title} </big> </p>
    <isse:getEntries parentId="{topic.id}">
      <c:forEach var="entry" items="{entries}">
        <p>
          
          <a href="entry.jsp?entry_id={entry.id}"> {entry.title} </a>
          <br / >
          <small> URL: <a href="{entry.url}"> {entry.url} </a> </small>
          <br / >
          {entry.description} &nbsp; &nbsp; &nbsp;
          <a href="entry.jsp?entry_id={entry.id}">more > </a>
        </p>
      </c:forEach>
    </isse:getEntries>
  </c:forEach>
</isse:getTopics>
```

getEntry Tag**Argument:**

<i>Argument</i>	<i>Description</i>	<i>Required/Optional</i>
entryId	Id of entry	required

Description:

`getEntry` returns a an Entry object where that entry is associated with the `entryId` provided. Conceptually, the returned collection would be equivalent to the collection with the following SQL:

```
SELECT * FROM entry
WHERE entry.id = entryId;
```

Tag body:

The tag body will expect an Entry object to be returned in the page-scoped variable `entry`. That object's attributes will be retrieved:

Example:

On the Entry Display page, the following code retrieves the `entry` object's attributes:

```
<table>
  <isse:getEntry entryId="{entry.id}">
    <tr><td> Name: </td><td> {entry.title} </td> </tr>
    <tr><td> URL: </td><td> {entry.url} </td></tr>
    <tr><td> Type: </td><td> {entry.type} </td></tr>
    <tr><td> Description: </td><td> {entry.description} </td></tr>
    <tr><td> Abstract: </td><td> {entry.abstract} </td></tr>
    <tr><td> Participants: </td><td> {entry.participants} </td></tr>
    <tr><td> Start Date: </td><td> {entry.startDate} </td></tr>
    <tr><td> Completion Date: </td><td> {entry.endDate} </td></tr>
    <tr><td> Last Modified Date: </td><td> {entry.lastModDate} </td></tr>
    <tr><td> Funders: </td> <td> {entry.funders} </td></tr>
    <tr><td> Geographic Location: </td><td> {entry.location} </td></tr>
    <tr><td> Themes and Topics: </td><td> {entry.topics} </td></tr>
    <tr><td> Related Projects: </td><td> {entry.relatedEntryId} </td></tr>
  </isse:getEntry>
</table>
```

searchEntries Tag**Arguments:**

<i>Argument</i>	<i>Description</i>	<i>Required/Optional</i>
searchString	String to search for (can include regexp)	required
dbField	Field to search	required
sortBy	Field to sort by	optional (default is no sort)

Description:

`searchEntries` returns a array of collection of Entry objects. This tag queries the specified database field (`dbField`) looking for the specified search parameter (`searchString`), and returns a collection of Entry objects that are sorted by the specified `sortBy` field. Conceptually, the returned collection would be equivalent to the collection with the following SQL:

```
SELECT * FROM dbField WHERE REGEXP "searchString" ORDER BY sortBy;
```

Tag body:

The tag body will expect a collection of Entry objects to be returned in the page-scoped variable of `entries`. That collection will be iterated upon in the tag body.

Example:

On the Project List, the following code iterates over a collection of `letters`, and then all the entries whose titles correspond to the letter passed as an argument. We assume we have a collection of `letters`, which will act as search parameters (`searchEntry`):

```
<c:forEach var="letter" items="{letters}">
  <tr><td> <big class="c">${letter}</big> </td></tr>
  <isse:searchEntries searchString="^{letter}" dbField="title"
  sortBy="title">
    <c:forEach var="entry" items="{entries}">
      <tr>
        <td>
          <a href="{entry.url}">${entry.title}</a>
        </td>
      </tr>
    </c:forEach>
  </isse:searchEntries>
</c:forEach>
```