

Example_Xml_Filter

This article is archived because it is not considered relevant for third-party developers creating commercial solutions today. The article is believed to be still valid for the original topic scope.



This script uses HTTP service access XML service, and filters content using XML filter.

WidSets Example

xml_filter.he

```
class
{
    const int    CMD_CANCEL = 1;
    const int    CMD_BACK   = 2;

    MenuItem    BACK    = new MenuItem(CMD_BACK, "Back");
    MenuItem    CANCEL  = new MenuItem(CMD_CANCEL, "Cancel");

    //randomly selected feed ("REST" api)
    String URL = "http://krumelur.jaiku.com/presence/xml";

    Flow m_flow;
    Prompt m_prompt;

    void startWidget()
    {
        setMinimizedView(createMinimizedView("viewMini", getStyle("default")));
    }

    Shell openWidget()
    {
        Flow flow = new Flow(getStyle("shell"));

        flow.setPreferredSize(-100, -100);
        requestData();
    }
}
```

Example_Xml_Filter

```
    return new Shell(m_flow = flow);
}

void addText(String name, String value)
{
    Label label = new Label(getStyle("name"), name);
    label.setFlags(VISIBLE|LINEFEED);
    label.setPreferredWidth(-100);
    m_flow.add(label);

    Text text = new Text(getStyle("value"), value);
    text.setFlags(VISIBLE|LINEFEED);
    text.setPreferredWidth(-100);
    m_flow.add(text);
}

void addPicture(String url)
{
    Picture pict = getPicture(url);
    pict.setStyle(getStyle("name"));
    m_flow.add(pict);
}

MenuItem getSoftKey(Shell shell, Component focused, int key)
{
    if (key == SOFTKEY_BACK) {
        return BACK;
    }
    return null;
}

void actionPerformed(Shell shell, Component source, int action)
{
    if (action == CMD_BACK) {
        popShell(shell);
    } else if (action == CMD_CANCEL) {
        closePrompt();
        //there's no actually a way to cancel on going
        //call() but atleast release the UI
    }
}

void requestData()
{
    Value arg = [
        "url" => URL
    ];
    call(null, "httpService", "get", arg, ok, nok);

    m_prompt = new Prompt(null, "Loading...", null, CANCEL);
    //prompt.setProgress(current, 10);
    m_prompt.push();

    void ok(Object state, Value ret)
    {
        closePrompt();
    }
}
```

Example_Xml_Filter

```
foreach (Value item : ret) {
    String key = item[0];
    String value = item[1];
    if ("avatar".equals(key)) {
        addPicture(value);
    } else {
        addText(key, value);
    }
}

void nok(Object state, String error)
{
    closePrompt();
    addText("Request failed", error);
}

void closePrompt()
{
    if (m_prompt != null) {
        m_prompt.pop();
        m_prompt = null;
    }
}
```

widget.xml

```
<?xml version="1.0" encoding="utf-8"?>

<widget spec_version="2.0">
  <info>
    <name>XML Filter</name>
    <version>1.3</version>
    <author>test</author>
    <clientversion>0.98</clientversion>
    <shortdescription>XML Filter example</shortdescription>
    <longdescription>XML Filter example demonstrationong content pulling from Jaiku</longdescription>
    <tags>helium example</tags>
  </info>

  <services>
    <service type="http" id="httpService">
      <filter id="digg"/>
    </service>
  </services>

  <filters>
    <filter id="digg">
      <list>
        <item name="nick">
          <xpath>/presence/user/@nick</xpath>
        </item>

        <item name="line">
          <xpath>/presence/line/text()</xpath>
        </item>
      </list>
    </filter>
  </filters>
</widget>
```

Example_Xml_Filter

```
<item name="location">
  <choice>
    <xpath>/presence/location/neighbourhood/text ()</xpath>
    <xpath>/presence/location/city/text ()</xpath>
    <xpath>/presence/location/country/text ()</xpath>
  </choice>
</item>

<item name="avatar">
  <xpath>/presence/user/@avatar</xpath>
</item>
</list>
</filter>
</filters>

<parameters>
  <parameter type="string" name="widgetname" description="Name of widget" editable="no" visible="no">
</parameters>

<resources>
  <code src="xml_filter.he"/>
  <stylesheet>
    shell {
      background: solid white;
      color: black;
    }

    name {
      color: black;
      font: medium bold;
      margin: 2sp 2sp 2sp 2sp;
    }

    value {
      color: black;
      font: medium plain;
      margin: 2sp 2sp 2sp 20sp;
    }

    minText {
      background: solid white;
      color: black;
      align: vcenter hcenter;
    }
  </stylesheet>
</resources>

<layout minimizedheight="2em">
  <view id="viewMini">
    <label class="minText">${widgetname}</label>
  </view>
</layout>
</widget>
```