

TelePort Modules

- HTTP
- Poll
- Socket



jpf.rpc Protocols

- POST
- HEADER
- REST
- SOAP
- XMLRPC
- JSON
- JPHP

jpf.rpc Properties

- isRPC
- useHTTP
- routeServer
- autoroute
- namedArguments
- protocol
- multicall
- useXML

jpf.rpc Methods

- addMethod
- setCallback
- revert
- purge
- load
- serialize
- unserialize

jpf.http Properties

- timeout

jpf.jtpp Methods

- toString
- saveCache
- loadCache
- getXml
- get
- cancel
- retry

Javeline IFDEF's

- \_\_WITH\_TELEPORT
- \_\_TP\_IFRAME
- \_\_TP\_SOCKET
- \_\_TP\_POLL
- \_\_TP\_RPC
- \_\_TP\_RPC\_HEADER
- \_\_TP\_RPC\_JPHP
- \_\_TP\_RPC\_JSON
- \_\_TP\_RPC\_POST
- \_\_TP\_RPC\_REST
- \_\_TP\_RPC\_SOAP
- \_\_TP\_RPC\_XMLRPC
- \_\_WITH\_HTTPCACHING

POST Example

```

Javeline Markup Language:
<j:rpc id="comm" protocol="POST" url=
eval="HOST_PATH+ '/subscribe.cgi'" method=
name="func" autoroute="true">
  <j:method name="login">
    <j:variable name="username" />
    <j:variable name="password" />
    <j:variable name="status" />
  </j:method>
</j:rpc>
Javascript:
<script>
  comm.login("ruben", "passwd", "online");
</script>
    
```

XML-RPC Example (advanced options)

```

Javeline Markup Language:
<j:rpc id="comm" protocol="XMLRPC"
url="subscribe.cgi">
  <j:global name="version" value="1.0" />
  <j:method name="login" type="global"
variable="hash" />
  <j:method name="getUsers" />
  <j:method name="saveUser" />
  <j:method name="removeUser" />
  <j:method name="getStates"
caching="true" receive="fillstates" />
  <j:method name="dolt" async="false" />
</j:rpc>
    
```

HTML Form handling with TelePort

```

JavaScript:
function getHttpRes(msg, state, extra){
  //handler code
}
HTML:
<form action="example.php" onsubmit="new jpf.post().submitForm(this, getHttpRes); return false;">
    
```

Callback Example

```

function rcvBooks(xmlData, state, extra){
  if(state != _RPC_SUCCESS__){
    if(state == HTTP_TIMEOUT_ && extra.retries < MAX_RETRIES)
      return extra.tpModule.retry(extra.id);
    else throw new Error(0, "Error");
  }
  alert(xmlData);
}
comm.setCallback("ItemSearch", rcvBooks);
    
```

Model Initialization

```

Model tag
<j:model get="proclnstr" />
    
```

SmartBinding Rules

```

Javeline Markup Language:
<j:smartbinding>
  <j:bindings>
    <j:load select="." get="proclnstr" />
    <j:insert select="." get="proclnstr" />
  </j:bindings>
  <j:actions>
    <j:rename select="." set="proclnstr" />
    <j:remove select="." set="proclnstr" />
  </j:actions>
</j:smartbinding>
    
```

Processing Instructions

```

Model Instruction
model="id"
model="id:xpath"
model="#component:select"
model="#component:select:xpath"
model="#component:choose"
model="#component:choose:xpath"
model="#component:::xpath"
model="<source>"
    
```

```

Get instruction
get="<source>|xpath|clear"
Gets result from a source. Then selects a subset using xpath. Clear sets whether the contents of the insertion point is cleared before appending the data.
    
```

```

Set instruction
set="<source>"
    
```

```

<source> syntax
Arguments also valid for normal model syntax
xpath:      xpath:/path/to/node
eval:       eval:5+10
method:     call:mName(xpath)
rpc:        rpc:comm.blah(xpath, call)
url:        url:http://example.com?x=xpath
cookie:     cookie:name.sub(xpath)
array:      ()
literal:   interpreted as javascript
    
```

```

Example: <j:load get="rpc.comm.getUserInfo
(xpath:@id, getCache(), 'extrainfo')|data/user|1" />
    
```

State Variables

- \_\_RPC\_SUCCESS\_\_
- \_\_RPC\_ERROR\_\_
- \_\_RPC\_TIMEOUT\_\_

Extra Object

- id
- userdata
- http
- url
- tpModule
- message
- retries

JML Attributes

- <j:rpc>
- id
- protocol
- url
- url-eval
- method-name
- autoroute
- multicall
- ns-name
- ns-url
- offline

- <j:method>
- name
- receive
- async
- export
- type
- variable
- lookup
- caching

- <j:variable>
- name
- value
- default
- encoded