Downloaded from: justpaste.it/34sp

```
<!
DOCTYPEhtml>
<
html>


<!--
  Copyright (c) 2012 The Native Client Authors. All rights reserved.
  Use of this source code is governed by a BSD-style license that can be
  found in the LICENSE file.
  -->
<
head>


<title>hello_tutorial</title>


<scripttype="text/javascript">
    HelloTutorialModule =
null;  // Global application object.
    statusText =
'NO-STATUS';

// Indicate load success.

function moduleDidLoad() {
      HelloTutorialModule = document.getElementById(
'hello_tutorial');
      updateStatus(
'SUCCESS');
      HelloTutorialModule.postMessage(
'hello saggu veere');
    }

// The 'message' event handler.  This handler is fired when the NaCl module

// posts a message to the browser by calling PPB_Messaging.PostMessage()

// (in C) or pp::Instance.PostMessage() (in C++).  This implementation
```

```
// simply displays the content of the message in an alert panel.

function handleMessage(message_event) {
      alert(message_event.data);
    }

// If the page loads before the Native Client module loads, then set the

// status message indicating that the module is still loading.  Otherwise,

// do not change the status message.

function pageDidLoad() {

if (HelloTutorialModule == null) {
        updateStatus(
'LOADING...');
      }
else {

// It's possible that the Native Client module onload event fired

// before the page's onload event.  In this case, the status message

// will reflect 'SUCCESS', but won't be displayed.  This call will

// display the current message.
        updateStatus();
      }
    }

// Set the global status message.  If the element with id 'statusField'

// exists, then set its HTML to the status message as well.

// opt_message The message test.  If this is null or undefined, then

// attempt to set the element with id 'statusField' to the value of

// |statusText|.

function updateStatus(opt_message) {
```

```
if (opt_message)
        statusText = opt_message;

var statusField = document.getElementById('status_field');

if (statusField) {
        statusField.innerHTML = statusText;
      }
    }

</script>
</
head>
<
bodyonload="pageDidLoad()">
<
h1>Native Client Module HelloTutorial</h1>
<
p>

<!-- Load the published .nexe.  This includes the 'nacl' attribute which
  shows how to load multi-architecture modules.  Each entry in the "nexes"
  object in the .nmf manifest file is a key-value pair: the key is the
  instruction set architecture ('x86-32', 'x86-64', etc.); the value is a URL
  for the desired NaCl module.
  To load the debug versions of your .nexes, set the 'nacl' attribute to the
  _dbg.nmf version of the manifest file.
  Note: Since this NaCl module does not use any real-estate in the browser,
  it's width and height are set to 0.
Note: The <EMBED> element is wrapped inside a <DIV>, which has both a 'load'
  and a 'message' event listener attached.  This wrapping method is used
  instead of attaching the event listeners directly to the <EMBED> element to
  ensure that the listeners are active before the NaCl module 'load' event
  fires.  This also allows you to use PPB_Messaging.PostMessage() (in C) or
  pp::Instance.PostMessage() (in C++) from within the initialization code in
  your NaCl module.
  -->

<divid="listener">

<scripttype="text/javascript">
```

```
var listener = document.getElementById('listener');
      listener.addEventListener(
'load', moduleDidLoad, true);
      listener.addEventListener(
'message', handleMessage, true);

</script>


<embedname="nacl_module"

id="hello_tutorial"

width=0height=0

src="hello_tutorial.nmf"

type="application/x-nacl"/>

</div>
</
p>
<
h2>Status</h2>
<
divid="status_field">NO-STATUS</div>
</
body>
</
html>
```