# Add data to a web page with JavaScript, WebKit, and Qt

Archived: This article is **archived** because it is not considered relevant for third-party developers creating commercial solutions today. If you think this article is still relevant, let us know by adding the template {{ReviewForRemovalFromArchive|user=~~~~|*write your reason here*}}.

The article is believed to be still valid for the original topic scope.

## Overview

This snippet demonstrates how you can add data to a web page with Qt via WebKit. We will use the article Gather data from web page with JavaScript, WebKit, and Qt as base for this snippet.

## Preconditions

- Qt is installed on your platform.
  - S60:
    - Download Qt release from here: Qt pre-release
    - Install Qt: How to Install Qt
    - Check this link for installation guide: How to install the package
  - Maemo:
    - More information about Qt on Maemo can be found here: Qt4 Maemo port
- You've read the article: Gather data from web page with JavaScript, WebKit, and Qt

## Code

Every time you see

```
/* ... */
```

there's code omitted from the source file for the sake of emphasising the code added to this snippet compared to the snippet this is based on.

### js/add-data.js

This code will be injected from Qt to the web page and it'll add user input to the web page as a new paragraph.

```
/**
  * Adds text p element with text content
  * after p element with id paragraph.
  */
Qt_QWET_add = function(string) {
  var p = document.getElementById("paragraph");
  var s = p.nextSibling;
  while(s.name != "" &&
        s.name == "p") {
    s = s.nextSibling;
  }
  var n = document.createElement("p");
  n.appendChild(document.createTextNode(string));
  var parent = s.parentNode;
  parent.insertBefore(n, s);
};
Qt_QWET_add("%1");
```


(C) Copyright Nokia 2014. All rights reserved.

Add the file to the resource file and include the resource file in your project file as explained in Using resources in Qt.

## src/qwet.h

The only things added to the header file in comparison to Gather data from web page with JavaScript, WebKit, and Qt are:

- `QLineEdit _text` - for text input
- `QString _addJS` - for injected JavaScript
- `addButtonClicked` slot
- `escapeCharacters` - helper method which escapes user input so that it can be used in generated JavaScript.

```
class QWET : public QMainWindow
{
    /* ... */

private slots:
    /* ... */
    void addButtonClicked();
private:
    /* ... */
    QPointer<QLineEdit> _text;

    QString _addJS;
    /* ... */
    QString escapeCharacters(const QString& string);
};

#endif // QWET_H
```

## src/qwet.cpp

Things which have changed compared to Gather data from web page with JavaScript, WebKit, and Qt:

- addition of another button to click to add data from Qt
- creation of `QHBoxLayout` to add the input widgets in
- the `addButtonClickedSlot` slot implementation
- `escapeCharacters` implementation

```
/* ... */
void QWET::setupUI()
{
    /* ... */

    /** Create a text input. */
    _text = new QLineEdit(this);

    /** Add the add text button. */
    QPushButton* addButton = new QPushButton("Add text.");
    connect(addButton, SIGNAL(clicked()),
            this, SLOT(addButtonClicked()));

    /**
      * Add the input widgets to own layout and add it to
      * main layout.
      */
    QHBoxLayout* input = new QHBoxLayout(this);
```

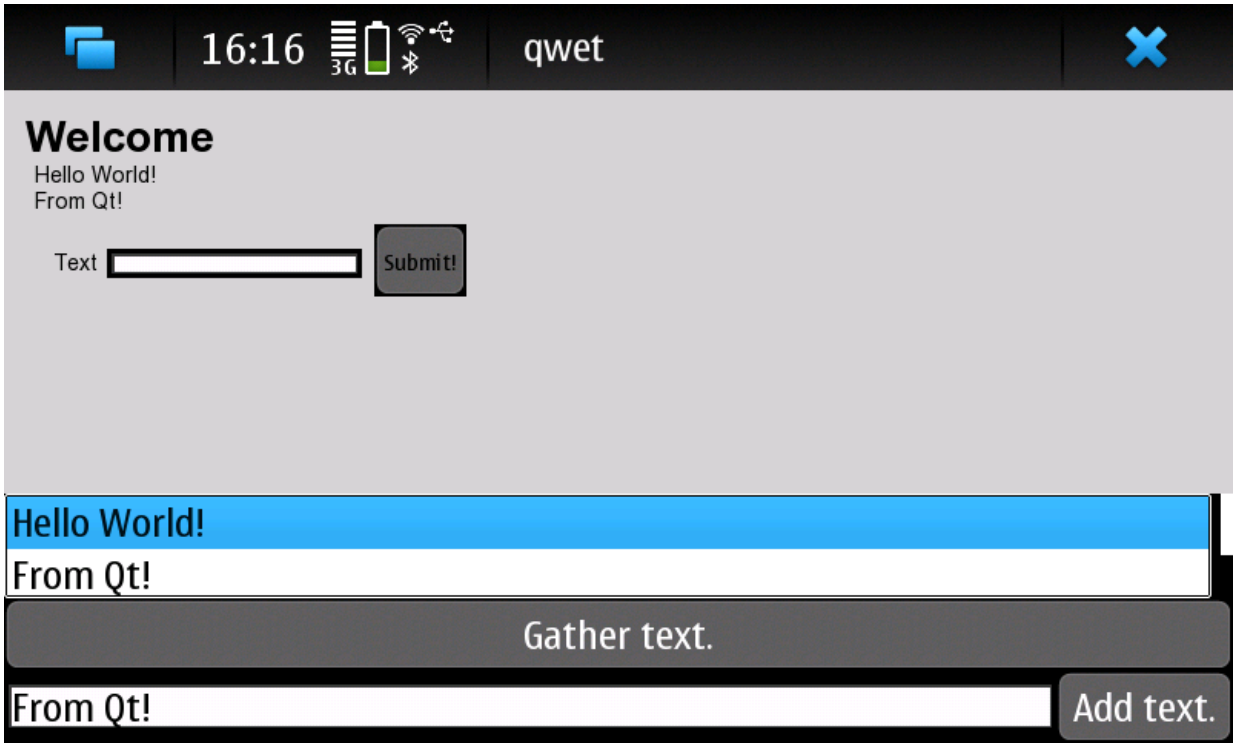

(C) Copyright Nokia 2014. All rights reserved.

```
    input->addWidget(_text);
    input->addWidget(addButton);
    _layout->addLayout(input);
}

/**
  * This slot adds a p element with content to the web page.
  */
void QWET::addButtonClicked()
{
    /** First time the add button is clicked, _addJS will be empty. */
    if(this->_addJS.isEmpty()) {
        /** Read the java script to be executed to a string. */
        this->_addJS = this->readFileToQString(QString(":/js/add-data.js"));
    }
    else {
        /** Second time we can reuse the same function again. */
        this->_addJS = QString("Qt_QWET_add(\"%1\");");
    }
    /** We'll have to escape the characters from the user input
      * so that we won't break the JavaScript we are generating. */
    QString string = this->escapeCharacters(this->_text->text());
    QString js = this->_addJS.arg(string);
    /** Run the java script on the web page. */
    this->_webView->page()->mainFrame()->evaluateJavaScript(js);
}

/* ... */

/**
  * Escapes the string from things which
  * aren't wanted to the JavaScript.
  */
QString QWET::escapeCharacters(const QString& string)
{
    QString rValue = QString(string);
    /** Assign \\ to backSlash */
    QString backSlash = QString(QChar(0x5c)).append(QChar(0x5c));
    /** Replace \ with \\ */
    rValue = rValue.replace('\\', backSlash);
    /** Assing \" to quote. */
    QString quote = QString(QChar(0x5c)).append(QChar(0x22));
    /** Replace " with \" */
    rValue = rValue.replace('"', quote);
    return rValue;
}
```

## Postconditions

You've now created this:


(C) Copyright Nokia 2014. All rights reserved.




## Complementary material

- File:Qwet.zip

## See also

- Display local web page with Qt WebKit
- Gather data from web page with JavaScript, WebKit, and Qt


(C) Copyright Nokia 2014. All rights reserved.