> Author: dailey.dai@openthinks.com

Network

QML URL

Image

```
import QtQuick 2.0
Image {
    source: "https://upload.wikimedia.org/wikipedia/sl/thumb/7/75/Qt-logo.svg/100px-Qt-logo.svg.png"
}
```

Qt.createComponent

```
Qt.createComponent("http://127.0.0.1/panel.qml");
```

Loader

```
import QtQuick 2.0
Item{
    Loader{
        source:"http://127.0.0.1/panel.qml";
    }
}
```

MediaPlayer

```
MediaPlayer{
    id:player;
    source:"http://127.0.0.1/12.mp3";
}
```

QQmlApplicationEngine

```
QQmlApplicationEngine engine;
engine.load(QUrl("http://127.0.0.1/main.qml"));
```

QQuickView

```
QQuickView viewer;
viewer.setResource(QUrl("http://127.0.0.1/main.qml"));
```

QML XMLHttpRequest

Attributes

readyState

status	name
0	Uninitialized
1	Open
2	Sent
3	Receiving
4	Loaded

responseText

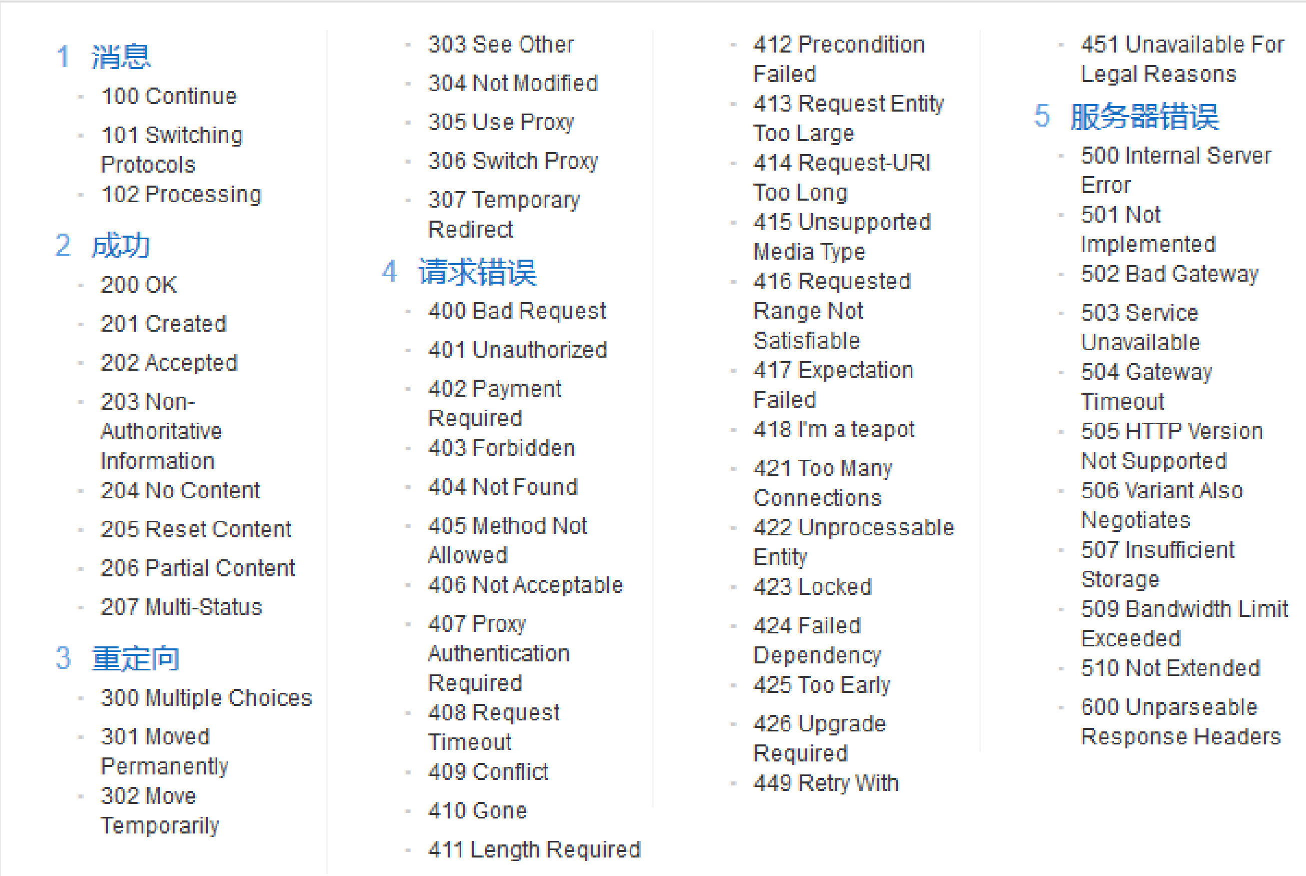
HTTP response body

responseXML

HTTP response XML DOM

status statusText

HTTP status code and status text



1 消息
- 100 Continue
- 101 Switching Protocols
- 102 Processing

2 成功
- 200 OK
- 201 Created
- 202 Accepted
- 203 Non-Authoritative Information
- 204 No Content
- 205 Reset Content
- 206 Partial Content
- 207 Multi-Status

3 重定向
- 300 Multiple Choices
- 301 Moved Permanently
- 302 Move Temporarily
- 303 See Other
- 304 Not Modified
- 305 Use Proxy
- 306 Switch Proxy
- 307 Temporary Redirect

4 请求错误
- 400 Bad Request
- 401 Unauthorized
- 402 Payment Required
- 403 Forbidden
- 404 Not Found
- 405 Method Not Allowed
- 406 Not Acceptable
- 407 Proxy Authentication Required
- 408 Request Timeout
- 409 Conflict
- 410 Gone
- 411 Length Required
- 412 Precondition Failed
- 413 Request Entity Too Large
- 414 Request-URI Too Long
- 415 Unsupported Media Type
- 416 Requested Range Not Satisfiable
- 417 Expectation Failed
- 418 I'm a teapot
- 421 Too Many Connections
- 422 Unprocessable Entity
- 423 Locked
- 424 Failed Dependency
- 425 Too Early
- 426 Upgrade Required
- 449 Retry With
- 451 Unavailable For Legal Reasons

5 服务器错误
- 500 Internal Server Error
- 501 Not Implemented
- 502 Bad Gateway
- 503 Service Unavailable
- 504 Gateway Timeout
- 505 HTTP Version Not Supported
- 506 Variant Also Negotiates
- 507 Insufficient Storage
- 509 Bandwidth Limit Exceeded
- 510 Not Extended
- 600 Unparseable Response Headers

onreadystatechange

`readyState` change callback

Methods

- open(method,url,async,username,password)
- setRequestHeader(name,value)
- send(body)
- abort()
- getResponseHeader(name)
- getAllResponseHeaders()

```
import QtQuick 2.0
import QtQuick.Controls 1.0
import QtQuick.Window 2.3

Rectangle{
    width: 400;height: 300;
    property var xmlhttp: null;
    TextEdit{
        id:content;
        anchors.fill: parent;
        anchors.bottomMargin: 20;
        textFormat: Text.RichText
        text: ""
    }
    Button{
        id:load;
        text:"Load";
        anchors.bottom: parent.bottom;
        anchors.left: parent.left;
        anchors.leftMargin: 2;
        onClicked: {
            loadHttpUrl();
        }
    }
    TextInput {
        id: url;
        anchors.bottom: load.bottom;
        anchors.left: load.right;
        anchors.leftMargin: 4;
        text: qsTr("https://cn.bing.com")
    }
    function onResultReady(){
        if(xmlhttp.readyState===4){
            console.log(xmlhttp.responseText)
            content.text=xmlhttp.responseText;
        }
    }

    function loadHttpUrl(){
        var sUrl = url.text;
        if(xmlhttp==null){
            xmlhttp= new XMLHttpRequest();
            xmlhttp.onreadystatechange =onResultReady;
        }
        if(xmlhttp.readyState === 0){
            xmlhttp.open("GET",sUrl,true);
            xmlhttp.send(null);
        }
    }
}
```

Qt C++ Network

- `QNetworkAccessManager`
- `QNetworkReply`
- `QNetworkRequest`
- `Qurl`

```
QNetworkAccessManager m_netmanager;
QNetworkReply * m_reply;
void onFinished();
int main(int argc, char *argv[])
{
QString strUrl("http://cn.bing.com");
QUrl qurl(strUrl);
QNetworkRequest req(qurl);
m_reply = m_netmanager.get(req);
connect(m_reply,SIGNAL(finished()),this,SLOT(onFinished()))
}
void onFinished()
{
    int status = m_reply->attribute(QNetworkRequest::HttpStatusCodeAttribute).toInt();
    if(status!=200){
        QByteArray data = m_reply->readAll();
    }
}
```