ECE4574 – Large-Scale SW Development for Engineering Systems
Lecture 7 – Network Programming; Frameworks

Creed Jones, PhD

Course Updates

- Sprint 1 starts on Monday
  - Project proposals will be graded by the end of the day on Thursday
- HW1 due this Friday, 11:59 PM
- Quiz 3 next Wednesday, September 20

Today's Objectives

Network Programming

- App-level network programming
- SSL/TLS
  - Public Key Infrastructure

Application Programming Interfaces

- What is an API
- How do we use an API

Frameworks

- What is a framework?
- Examples of common frameworks

SOCKETS PROGRAMMING

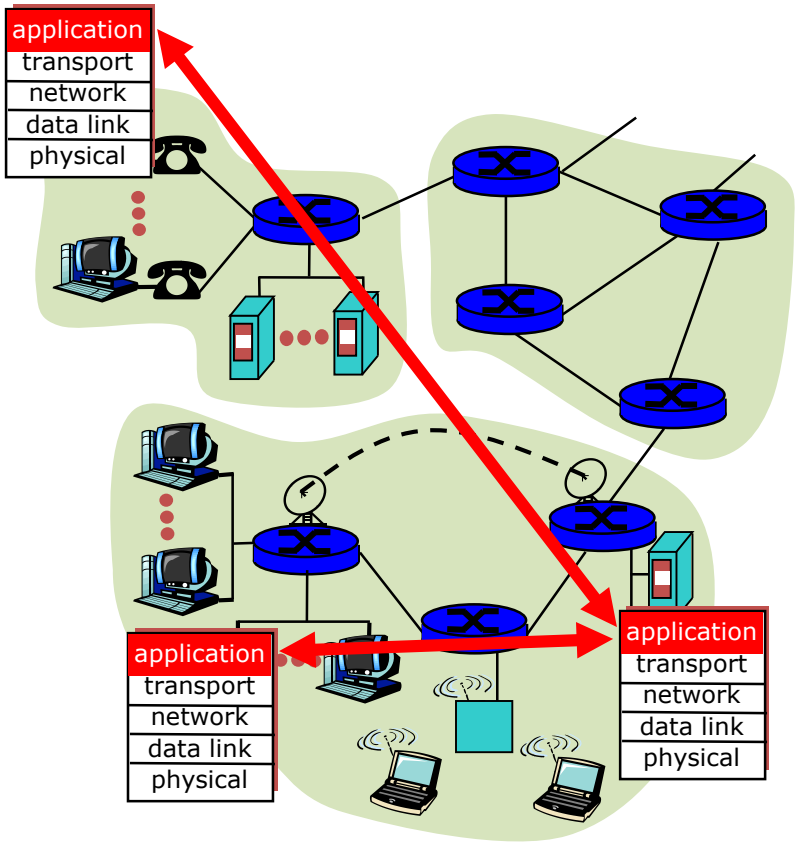
Creating a network application

Write programs that

- run on different end systems and
- communicate over a network.
- e.g., Web: Web server software communicates with browser software

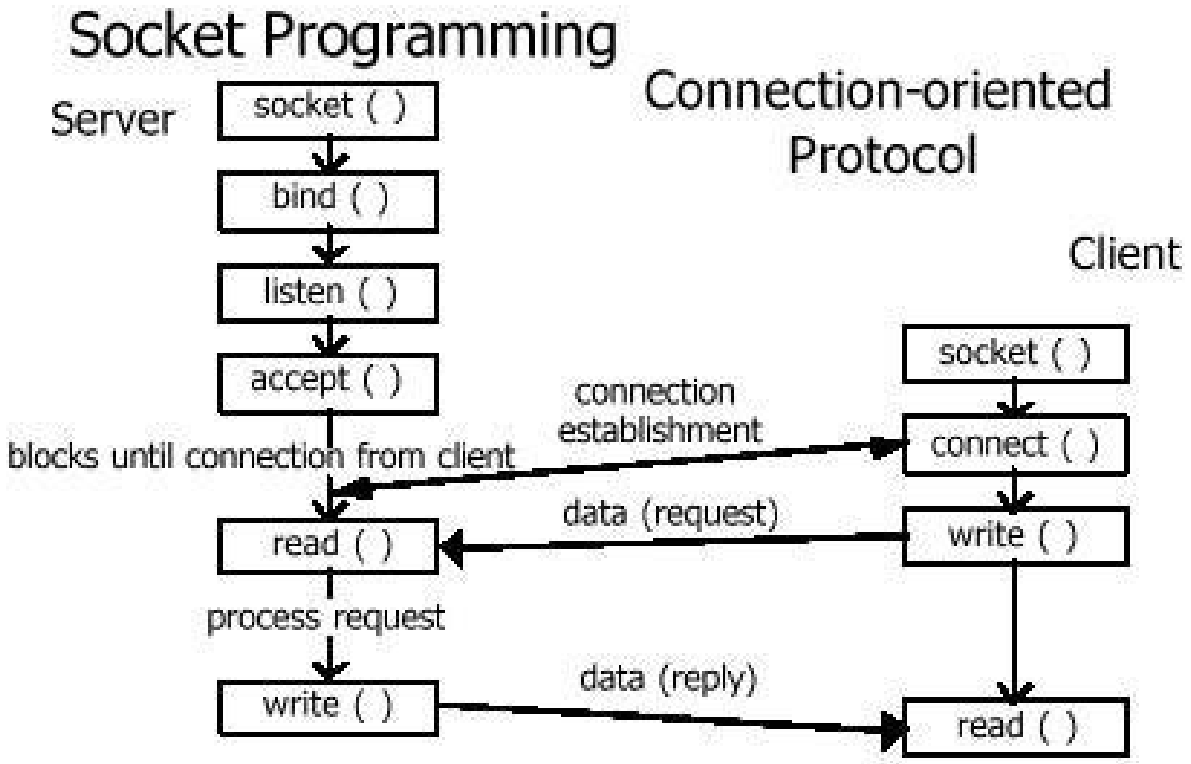
Less software is written for devices in network core

- network core devices do not run user application code
- application on end systems allows for rapid app development, propagation



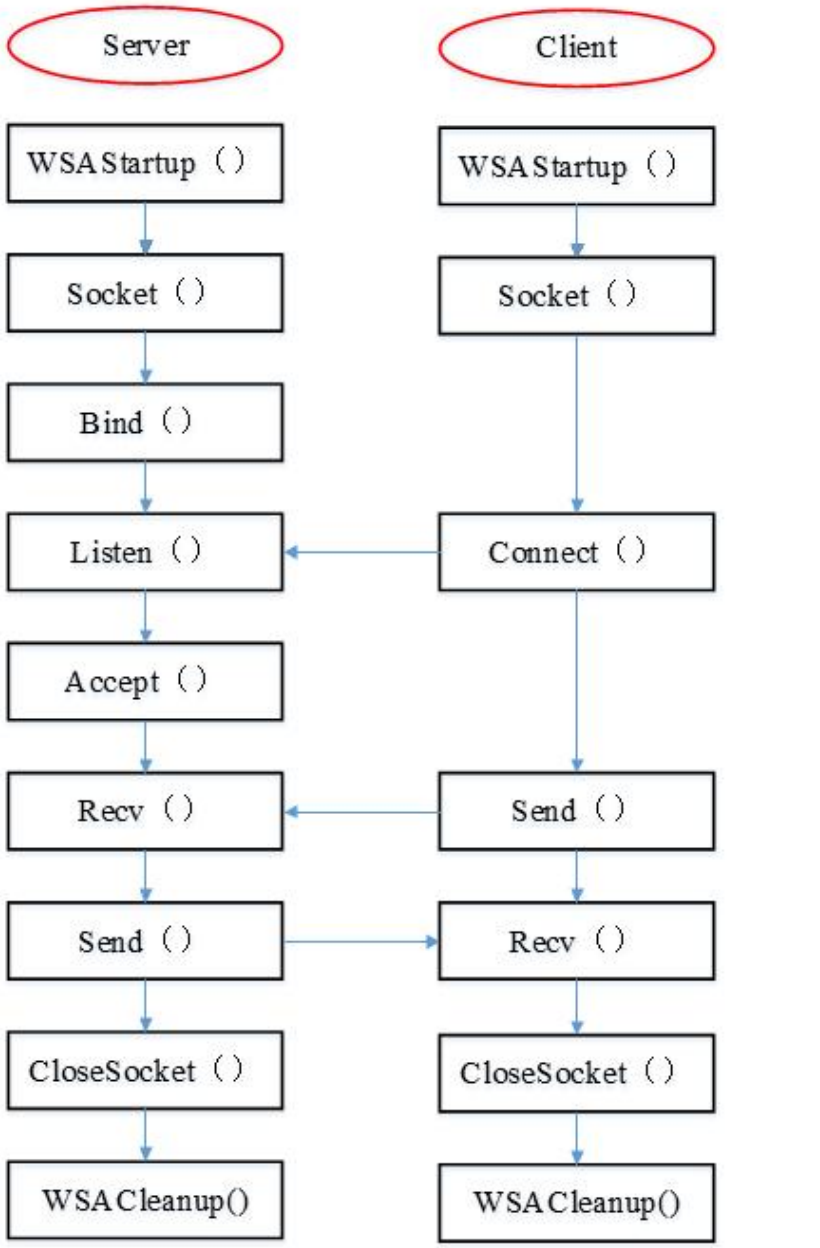
A socket is a way of connecting two processes on (typically) different machines on the network

- Socket A listens for a message on a particular port, on a particular IP address
- Socket B sends a message to the chosen port, on the IP address
- Socket A furnishes the message to the program, and further communication can take place in similar fashion
- Sockets require setup and connection



WinSock is the standard sockets implementation on Windows machines

- There's an API for us to write programs
  - Applications Programming Interface
- There's also an SPI for developers to write their own transport layer
  - Service Provider Interface
- On Linux, the sockets API is part of the standard OS programming interface
- I will show an example, communicating with a web server
  - BUT, the edge machines can both be custom code
  - We can define a port number and write a system that runs on two machines on the web



Demonstration of a Winsock app in Visual Studio

- This application uses HTTP to talk to a web server
- I had to choose carefully since very few web sites still use HTTP, without traffic encryption
- The URL being accessed is www.geocities.com
- The complete HTTP request is:

```
GET / HTTP/1.1
Host: www.geocities.com
Connection: keep-alive
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla /5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.121 Safari/537.36
Accept: text/html, application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\n\
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9
```

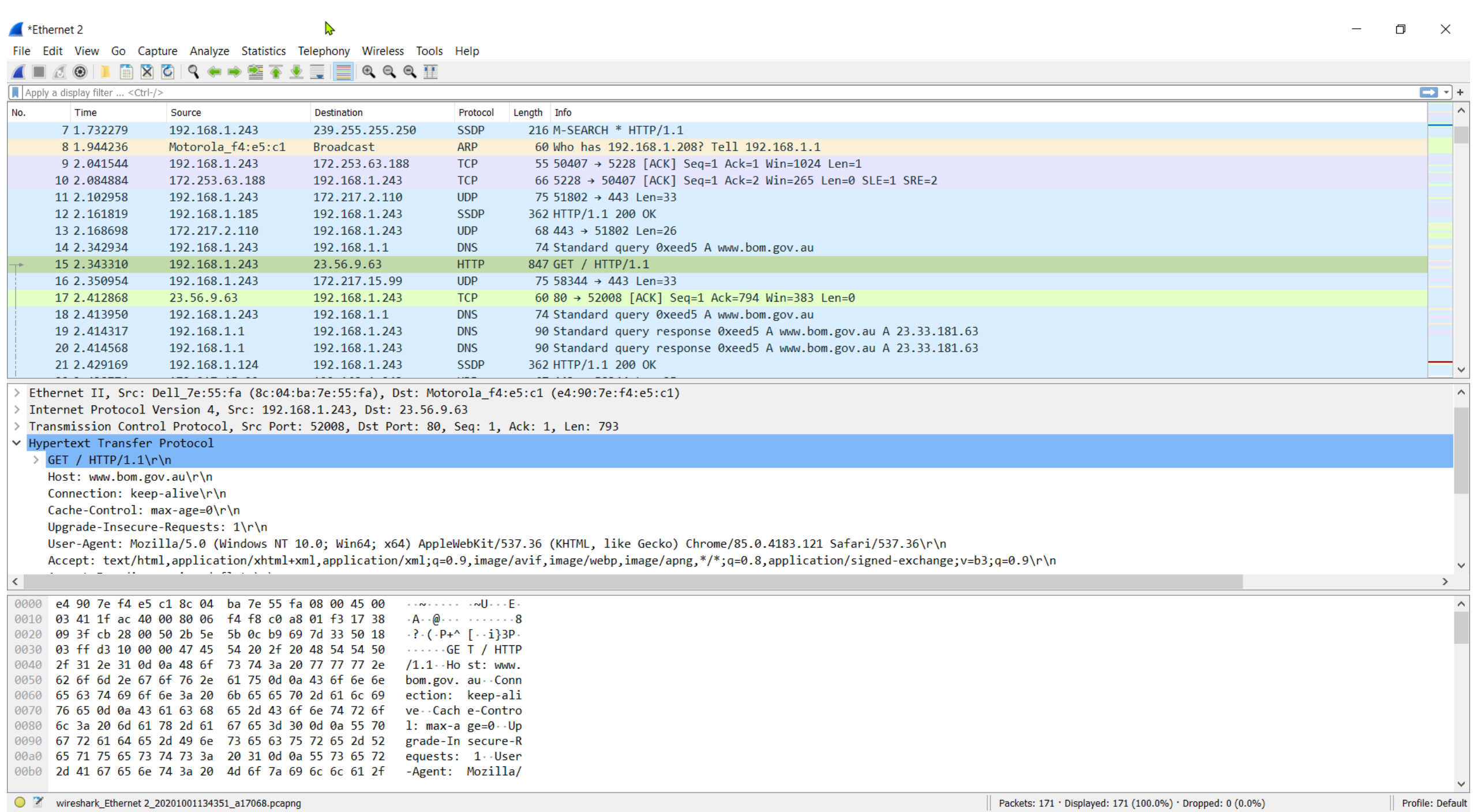
Where did I get the complete HTTP request, with all of the optional tags? I cheated – I used a *packet sniffer* to discover what my web browser sent when visiting the site

- A packet sniffer captures and displays all of the traffic on the network
- By running it on my own machine, I see all packets going to/from my machine – as well as everything else on my local network

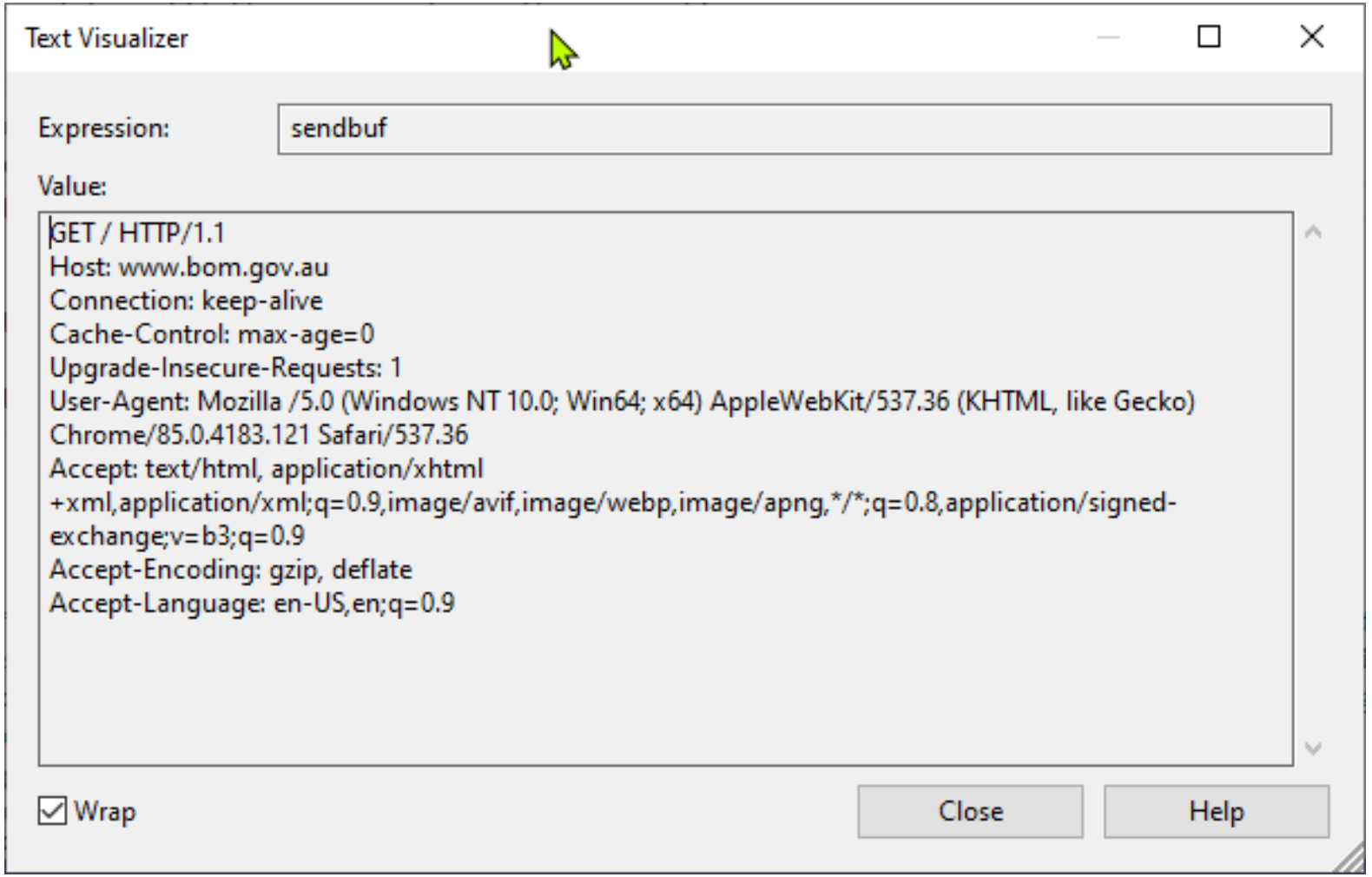
- I just copied the raw traffic transmitted when my browser issued an HTTP request, and pasted it into my code

- I did this because modern web servers require the right things in terms of "languages accepted", "formats accepted" and so on.

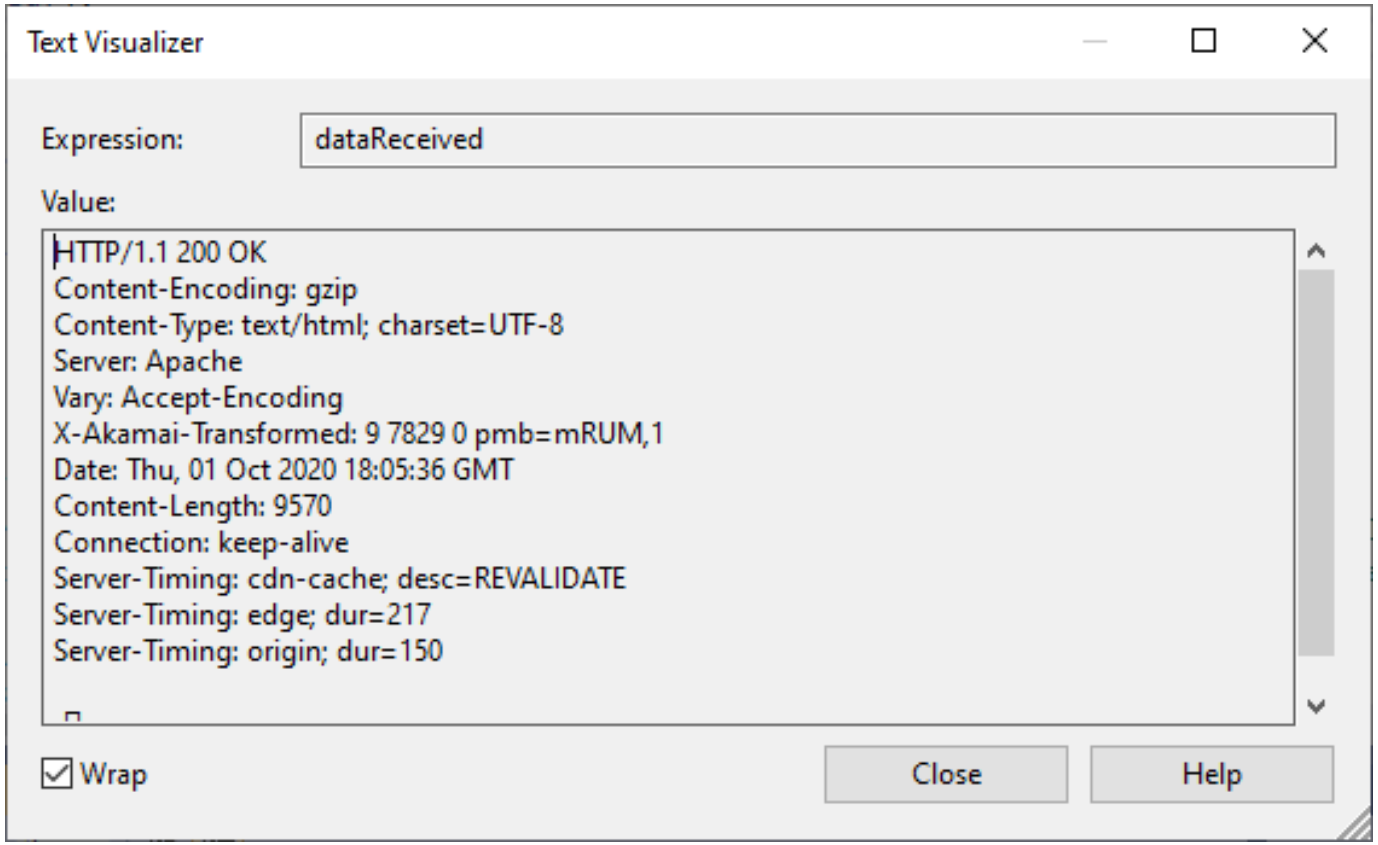
- I use WireShark - https://www.wireshark.org/


*Ethernet 2
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
No. Time Source Destination Protocol Length Info
7 1.732279 192.168.1.243 239.255.255.250 SSDP 216 M-SEARCH * HTTP/1.1
8 1.944236 Motorola_f4:e5:c1 Broadcast ARP 60 Who has 192.168.1.208? Tell 192.168.1.1
9 2.041544 192.168.1.243 172.253.63.188 TCP 55 50407 → 5228 [ACK] Seq=1 Ack=1 Win=1024 Len=1
10 2.084884 172.253.63.188 192.168.1.243 TCP 66 5228 → 50407 [ACK] Seq=1 Ack=2 Win=265 Len=0 SLE=1 SRE=2
11 2.102958 192.168.1.243 172.217.2.110 UDP 75 51802 → 443 Len=33
12 2.161819 192.168.1.185 192.168.1.243 SSDP 362 HTTP/1.1 200 OK
13 2.168698 172.217.2.110 192.168.1.243 UDP 68 443 → 51802 Len=26
14 2.342934 192.168.1.243 192.168.1.1 DNS 74 Standard query 0xeed5 A www.bom.gov.au
15 2.343310 192.168.1.243 23.56.9.63 HTTP 847 GET / HTTP/1.1
16 2.350954 192.168.1.243 172.217.15.99 UDP 75 58344 → 443 Len=33
17 2.412868 23.56.9.63 192.168.1.243 TCP 60 80 → 52008 [ACK] Seq=1 Ack=794 Win=383 Len=0
18 2.413950 192.168.1.243 192.168.1.1 DNS 74 Standard query 0xeed5 A www.bom.gov.au
19 2.414317 192.168.1.1 192.168.1.243 DNS 90 Standard query response 0xeed5 A www.bom.gov.au A 23.33.181.63
20 2.414568 192.168.1.1 192.168.1.243 DNS 90 Standard query response 0xeed5 A www.bom.gov.au A 23.33.181.63
21 2.429169 192.168.1.124 192.168.1.243 SSDP 362 HTTP/1.1 200 OK
Ethernet II, Src: Dell_7e:55:fa (8c:04:ba:7e:55:fa), Dst: Motorola_f4:e5:c1 (e4:90:7e:f4:e5:c1)
Internet Protocol Version 4, Src: 192.168.1.243, Dst: 23.56.9.63
Transmission Control Protocol, Src Port: 52008, Dst Port: 80, Seq: 1, Ack: 1, Len: 793
Hypertext Transfer Protocol
GET / HTTP/1.1\r\n
Host: www.bom.gov.au\r\n
Connection: keep-alive\r\n
Cache-Control: max-age=0\r\n
Upgrade-Insecure-Requests: 1\r\n
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/85.0.4183.121 Safari/537.36\r\n
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image/apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9\r\n
0000 e4 90 7e f4 e5 c1 8c 04 ba 7e 55 fa 08 00 45 00 ··~····· ·~U···E·
0010 03 41 1f ac 40 00 80 06 f4 f8 c0 a8 01 f3 17 38 ·A··@··· ·······8
0020 09 3f cb 28 00 50 2b 5e 5b 0c b9 69 7d 33 50 18 ·?·(·P+^ [··i}3P·
0030 03 ff d3 10 00 00 47 45 54 20 2f 20 48 54 54 50 ····GE T / HTTP
0040 2f 31 2e 31 0d 0a 48 6f 73 74 3a 20 77 77 77 2e /1.1··Ho st: www.
0050 62 6f 6d 2e 67 6f 76 2e 61 75 0d 0a 43 6f 6e 6e bom.gov. au··Conn
0060 65 63 74 69 6f 6e 3a 20 6b 65 65 70 2d 61 6c 69 ection: keep-ali
0070 76 65 0d 0a 43 61 63 68 65 2d 43 6f 6e 74 72 6f ve··Cach e-Contro
0080 6c 3a 20 6d 61 78 2d 61 67 65 3d 30 0d 0a 55 70 l: max-a ge=0··Up
0090 67 72 61 64 65 2d 49 6e 73 65 63 75 72 65 2d 52 grade-In secure-R
00a0 65 71 75 65 73 74 73 3a 20 31 0d 0a 55 73 65 72 equests: 1··User
00b0 2d 41 67 65 6e 74 3a 20 4d 6f 7a 69 6c 6c 61 2f -Agent: Mozilla/
wireshark_Ethernet 2_20201001134351_a17068.pcapng
Packets: 171 · Displayed: 171 (100.0%) · Dropped: 0 (0.0%)
Profile: Default

Here is my HTTP request, as sent by the application

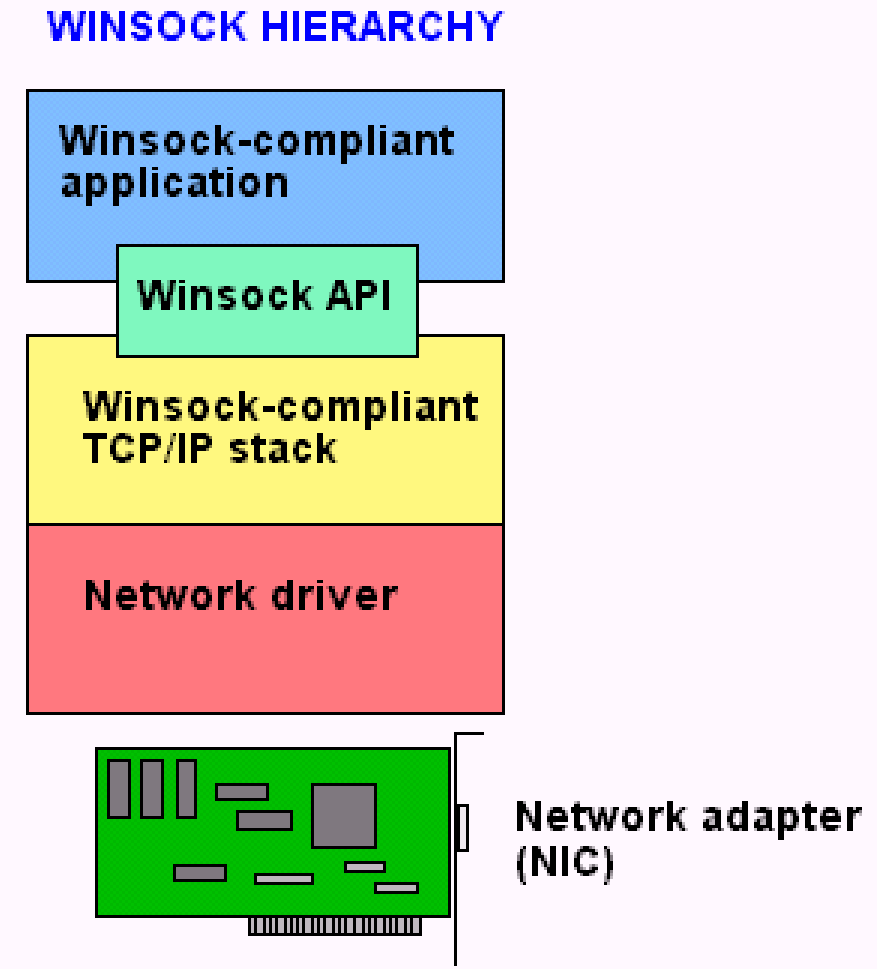


… and here is the response from the server at http:///bom.gov.au



APP-LEVEL NETWORK PROGRAMMING

When we use higher-level application programming interfaces like QNetworkAccessManager, they sit on top of, and use, the sockets API to access network resources



In Qt, there are several classes used to support simple HTTP and HTTPS web requests

- QNetworkAccessManager
  - There is one of these in an app using HTTP networking
  - It handles configuration and queueing of requests/results
  - Supports both HTTP and HTTPS, FTP, …
- QNetworkRequest
  - Contains a single request: URL, tokens, command, etc…
- QNetworkReply
  - The URL and headers of the network request, along with the data received in reply
  - The initial part of the data contains the HTTP reply code

```
#include <QCoreApplication>
#include "httpstalker.h"

int main(int argc, char *argv[]) {
    QCoreApplication a(argc, argv);
    HTTPSTalker talkObj;
    talkObj.tryHTTPS(QString("https://www.google.com"));
    return a.exec();
}
```

==========

```
// httpstalker.h
#include <QObject>
#include <QTCore>
#include <QNetworkAccessManager>
#include <QNetworkReply>

class HTTPSTalker : public QObject {
Q_OBJECT
    QNetworkAccessManager *manager;
public:
    HTTPSTalker();
    void tryHTTPS(QString);
public slots:
    void replyFinished(QNetworkReply*);
};
```

```
#include "httpstalker.h"

HTTPSTalker::HTTPSTalker() {
    manager = new QNetworkAccessManager();
}

void HTTPSTalker::tryHTTPS(QString hostName) {
    QNetworkRequest request;
    QNetworkReply *reply = NULL;
    connect(manager, SIGNAL(finished(QNetworkReply*)),
          this, SLOT(replyFinished(QNetworkReply*)));

    QSslConfiguration config =
          QSslConfiguration::defaultConfiguration();
    config.setProtocol(QSsl::TlsV1_2);
    request.setSslConfiguration(config);
    request.setUrl(QUrl(hostName));
    request.setHeader(QNetworkRequest::ServerHeader,
          "application/json");
    manager->get(request);
    qDebug() << request.url();
}

void HTTPSTalker::replyFinished(QNetworkReply* reply) {
    qDebug() << "REPLY: " << reply->error();
    qDebug() << reply->readAll();
}
```

```
#include <QCoreApplication>
#include "

int main(i
    QCoreA
    HTTPST
    talkOb
    return
}
==========

// httpsta
#include <
#include <
#include <
#include <

class HTTP
Q_OBJECT
    QNetwo
public:
    HTTPST
    void t
public slo
    void r
};
```

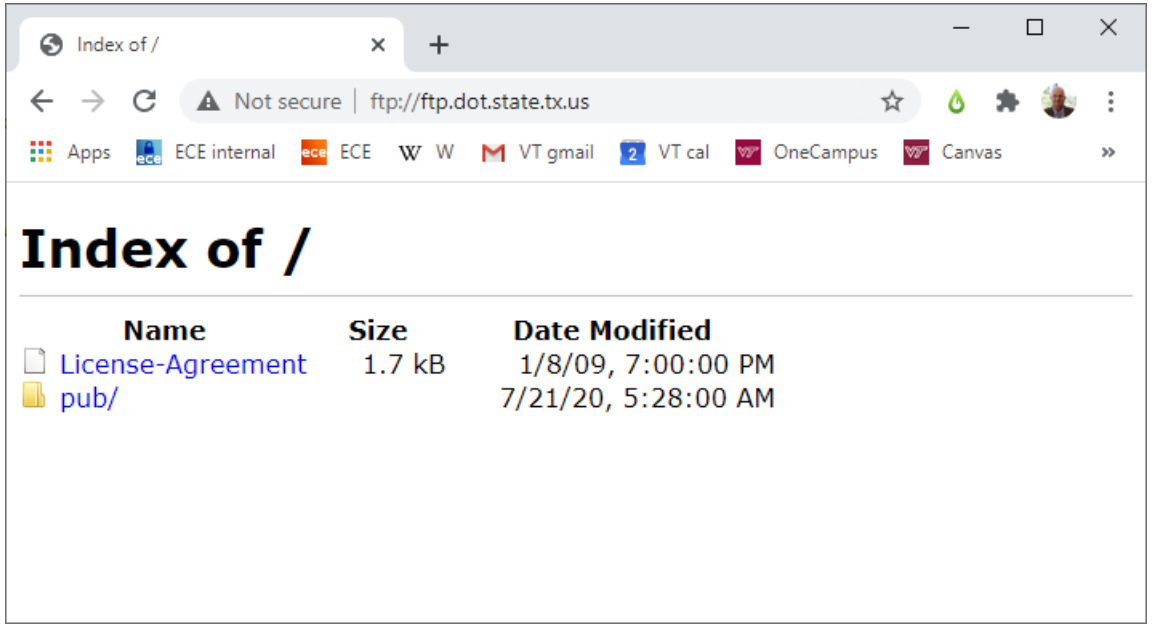
```
#include "httpstalker.h"

:HTTPSTalker() {
= new QNetworkAccessManager();


lker::tryHTTPS(QString hostName) {
kRequest request;
kReply *reply = NULL;
manager, SIGNAL(finished(QNetworkReply*)),
his, SLOT(replyFinished(QNetworkReply*)));

figuration config =
SslConfiguration::defaultConfiguration();
setProtocol(QSsl::TlsV1_2);
setSslConfiguration(config);
setUrl(QUrl(hostName));
setHeader(QNetworkRequest::ServerHeader,
application/json");
->get(request);
 << request.url();


lker::replyFinished(QNetworkReply* reply) {
 << "REPLY: " << reply->error();
 << reply->readAll();
```

D:\Qt\Qt5.12.9\Tools\QtCreator\bin\qtcr...

```
QUrl("https://www.google.com")
REPLY:  QNetworkReply::NoError
"<!doctype html><html itemscope=\"\" itemtype=\"http://schema.
org/WebPage\" lang=\"en\"><head><meta content=\"Search the wor
ld's information, including webpages, images, videos and more.
 Google has many special features to help you find exactly wha
t you're looking for.\" name=\"description\"><meta content=\"n
oodp\" name=\"robots\"><meta content=\"text/html; charset=UTF-
8\" http-equiv=\"Content-Type\"><meta content=\"/images/brandi
ng/googleg/1x/googleg_standard_color_128dp.png\" itemprop=\"im
age\"><title>Google</title><script nonce=\"AjRj42uoctPG5+P9MUM
eYQ==\">(function(){window.google={kEI:'9UZ7X_SPFfvDytMP1uuP0A
0',kEXPI:'0,18167,183992,3,1151585,5662,730,224,5105,206,3204,
10,1226,364,925,574,612,91,114,383,246,5,1354,648,652,2799,315
,3,66,308,145,321,211,270,2,10,869,112,452,320,89,324,66,726,9
3,143,356,1118466,1197740,543,302679,26305,13677,4855,32692,16
114,28684,9188,8384,4858,1362,9290,3029,2815,1924,11033,1808,4
020,978,7932,5296,2974,873,38,4162,6422,1142,13386,4517,1398,1
379,919,2277,8,2796,1593,1279,2212,530,149,1103,842,515,1466,4
,52,4258,108,204,1135,1,3,2063,606,2025,1731,43,144,377,1947,2
229,93,328,1284,17,2926,2247,1812,1787,3227,2846,6,6068,6286,4
454,642,1475,4659,1743,4928,108,3407,908,2,941,5012,1387,6080,
1,1706,1138,432,3,1546,44,1,820,1,4624,148,3502,2488,2252,5733
,4,1528,1696,608,1236,271,874,284,121,1860,2393,74,462,40,934,
281,52,214,762,1401,464,459,1555,4067,1035,1316,3,2108,175,997
,1426,69,305,1811,499,1514,1297,1753,690,1968,239,252,3673,78,
57,462,912,564,681,437,32,470,831,1410,1143,138,3095,990,52,17
75,1009,246,256,1219,994,850,80,368,1,98,908,638,37,109,519,3,
826,46,84,360,115,2,567,916,156,2730,610,38,954,754,615,2,53,4
```

ftp is the File Transfer Protocol – used for connecting to file servers (that support ftp) for upload/download

- ftp uses TCP, so data transfer is reliable
- you can send ftp client requests from a browser, or ftp clients like FileZilla
- ftps establishes a secure connection just as HTTPS does

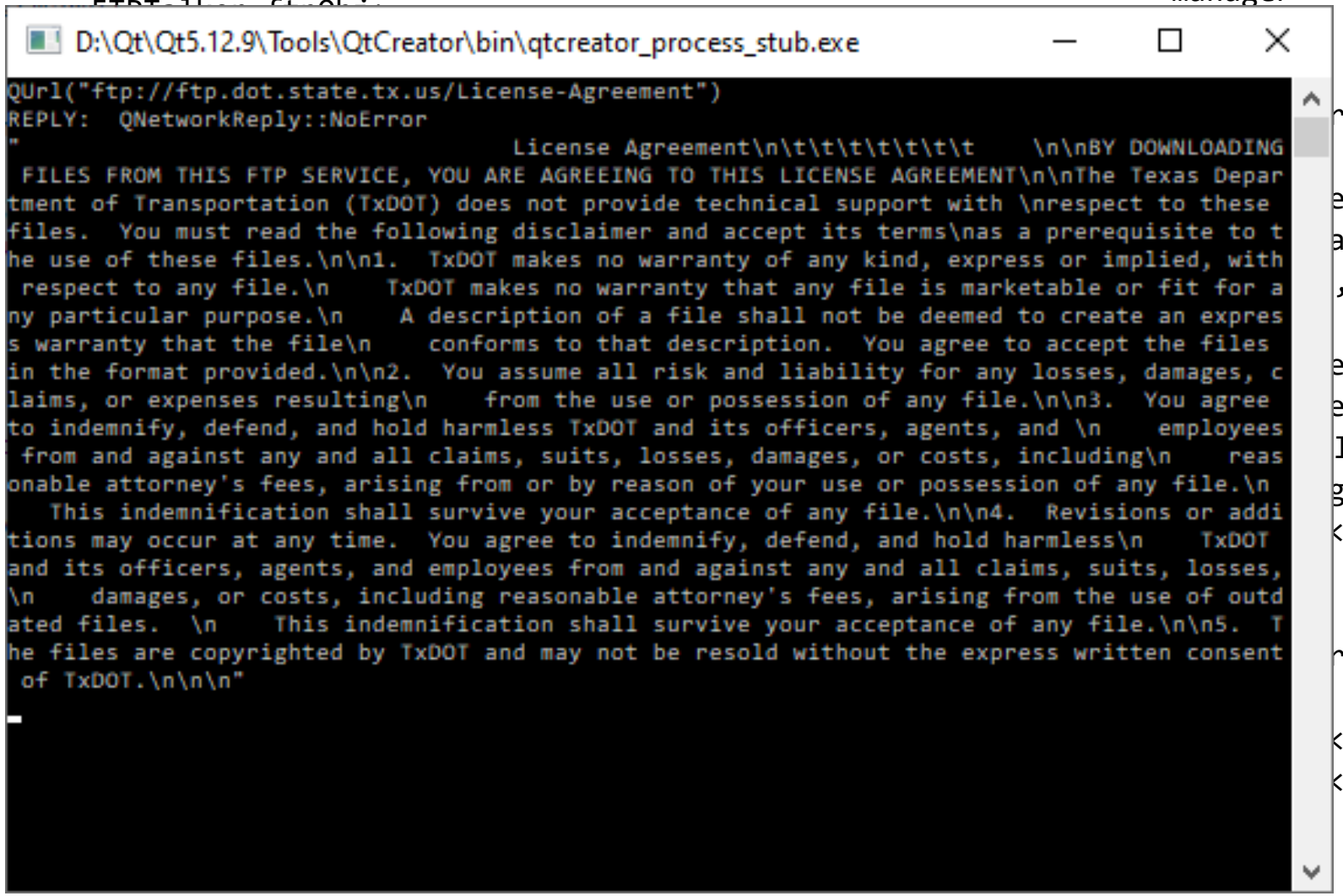


```
#include <QCoreApplication>
#include "ftptalker.h"

int main(int argc, char *argv[]) {
    QCoreApplication a(argc, argv);
    FTPTalker ftpObj;
    ftpObj.tryFTP(QString(
         "ftp://ftp.dot.state.tx.us/License-
Agreement"));
    return a.exec();
}
=========================================
// ftptalker.h

#include <QObject>

#include <QNetworkAccessManager>
#include <QNetworkReply>

class FTPTalker : public QObject {
Q_OBJECT
    QNetworkAccessManager *manager;
public:
    FTPTalker();
    void tryFTP(QString);

public slots:
    void replyFinished(QNetworkReply*);
};
```

```
#include "ftptalker.h"

FTPTalker::FTPTalker()
{
    manager = new QNetworkAccessManager();
}

void FTPTalker::tryFTP(QString hostName)
{
    QNetworkRequest request;
    connect(manager, SIGNAL(finished(QNetworkReply*)),
         this, SLOT(replyFinished(QNetworkReply*)));

    request.setUrl(QUrl(hostName));
    request.setHeader(QNetworkRequest::ServerHeader,
         "application/json");
    manager->get(request);
    qDebug() << request.url();
}

void FTPTalker::replyFinished(QNetworkReply* reply)
{
    qDebug() << "REPLY: " << reply->error();
    qDebug() << reply->readAll();
}
```

```
#include <QCoreApplication>
#include "ftptalker.h"

int main(int argc, char *argv[]) {
    QCoreApplication a(argc, argv);
```

```
#include "ftptalker.h"

FTPTalker::FTPTalker()
{
    manager = new QNetworkAccessManager();


r::tryFTP(QString hostName)

equest request;
anager, SIGNAL(finished(QNetworkReply*)),
, SLOT(replyFinished(QNetworkReply*)));

etUrl(QUrl(hostName));
etHeader(QNetworkRequest::ServerHeader,
lication/json");
get(request);
<< request.url();


r::replyFinished(QNetworkReply* reply)

<< "REPLY: " << reply->error();
<< reply->readAll();
```



VIRGINIA TECH

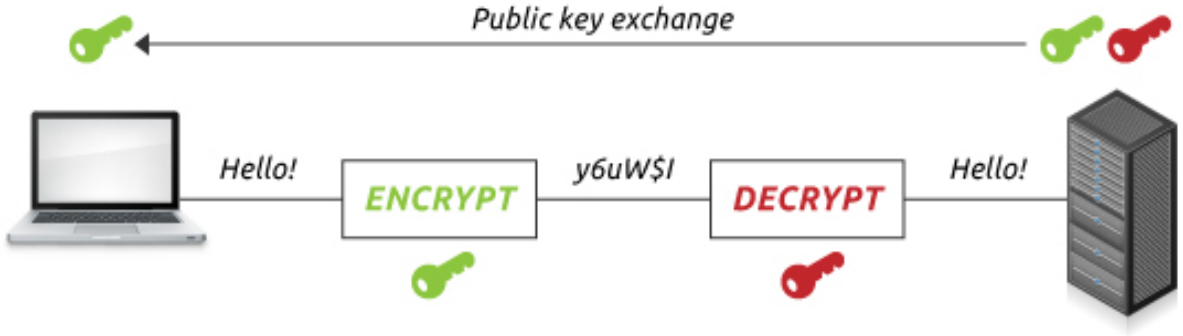
BRADLEY DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

WEB TRAFFIC SECURITY - SSL/TLS

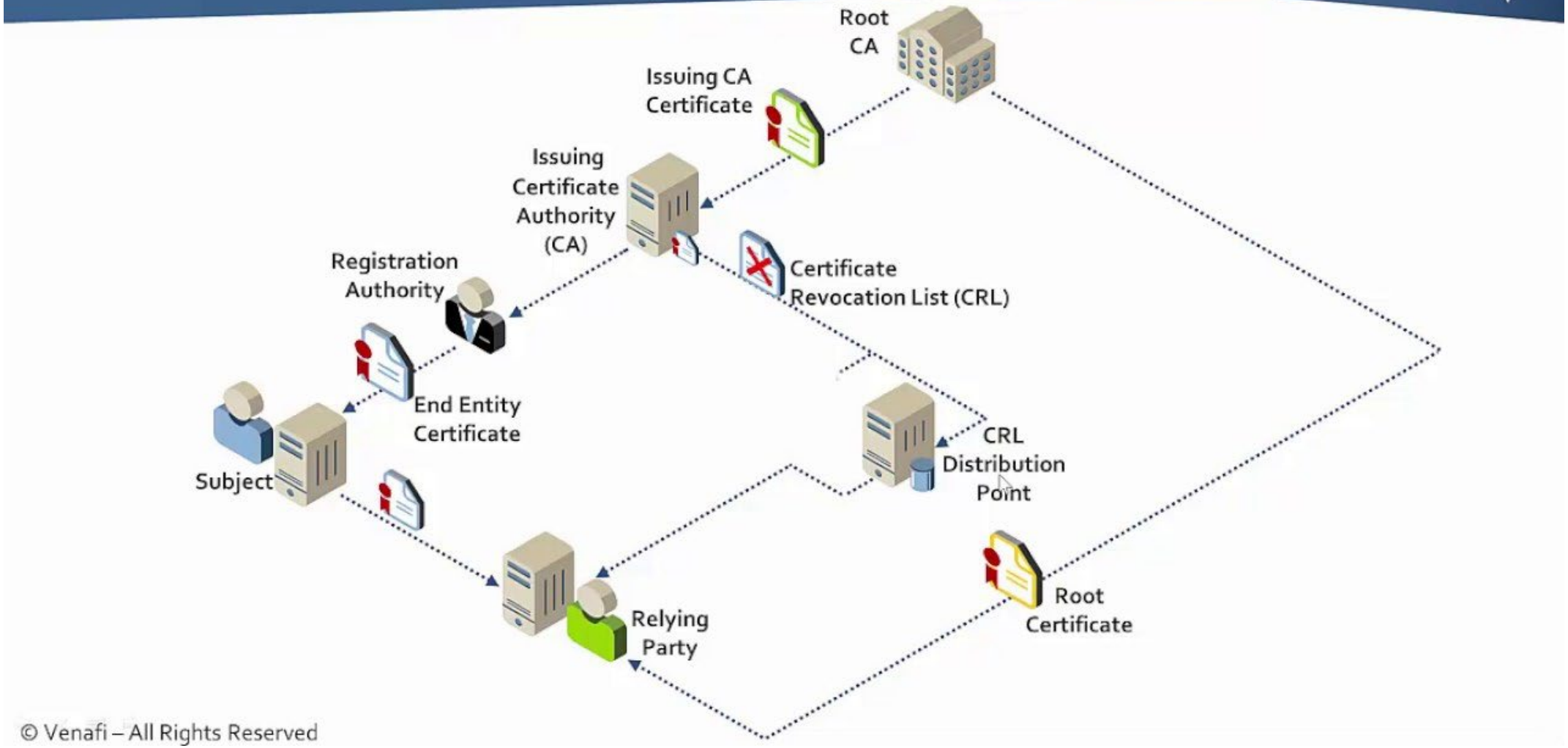
SSL (Secure Sockets Layer) and TLS (Transport Layer Security) encrypt application data before using IP to send it over the web

Though SSL was deprecated in 2015, we still say "SSL" – though it's usually the TLS protocol being used

1. Client says "I want to connect – here are ciphers I support"
2. Server says "let's use this one" and sends a certificate with its public key
3. Client confirms the certificate, so the key can be trusted
4. Client creates a session key, encrypts it with the public key, and sends it to server
5. The session key is now used to encrypt traffic for that session



Public Key Infrastructure (PKI)



https://venafi.com/platform/trust-protection-platform

APIS AND FRAMEWORKS

What is an API?

What is an API?

- "a set of routines, protocols, and tools for building software applications."

What is an API?

- "a set of routines, protocols, and tools for building software applications."
- "a set of programming instructions and standards for accessing a Web-based software application or Web tool. A software company releases its API to the public so that other software developers can design products that are powered by its service."

What is an API?

- "a set of routines, protocols, and tools for building software applications."
- "a set of programming instructions and standards for accessing a ~~Web-based~~ software application or Web tool. A software company releases its API to the public so that other software developers can design products that are powered by its service."

What is an API?

- "a set of routines, protocols, and tools for building software applications."
- "a set of programming instructions and standards for accessing a ~~Web-based~~ software application or Web tool. A software company releases its API to the public so that other software developers can design products that are powered by its service."
- "sets of requirements that govern how one application can talk to another."

What is an API?

- "a set of routines, protocols, and tools for building software applications."
- "a set of programming instructions and standards for accessing a ~~Web-based~~ software application or Web tool. A software company releases its API to the public so that other software developers can design products that are powered by its service."
- "~~sets of requirements that govern how one application can talk to another~~."

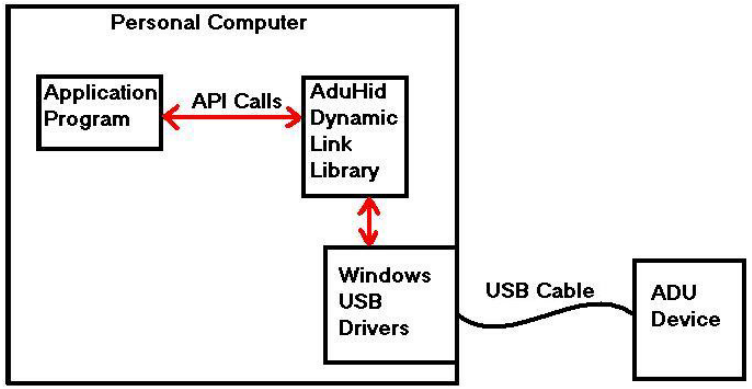
What is an API?

- "a set of routines, protocols, and tools for building software applications."
- "a set of programming instructions and standards for accessing a ~~Web-based~~ software application or Web tool. A software company releases its API to the public so that other software developers can design products that are powered by its service."
- "~~sets of requirements that govern how one application can talk to another~~."
- "An API specifies a software component in terms of its operations, their inputs and outputs and underlying types. Its main purpose is to define a set of functionalities that are independent of their respective implementation, allowing both definition and implementation to vary without compromising each other."

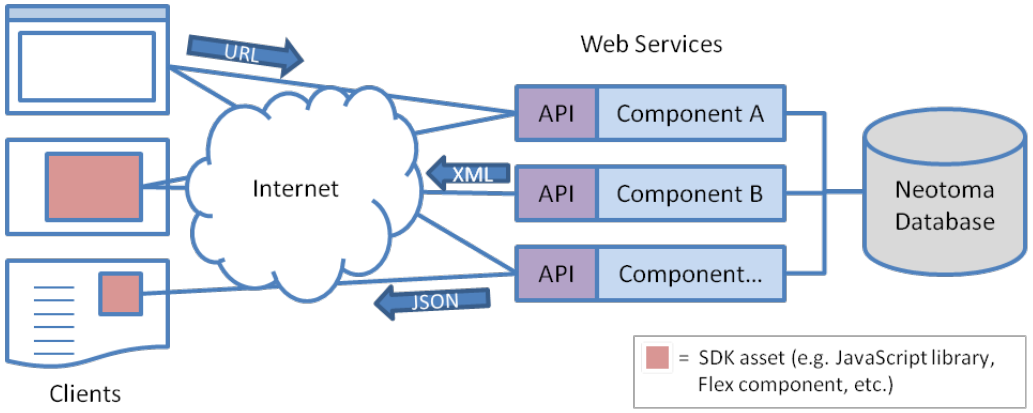
What is an API?

- "a set of routines, protocols, and tools for building software applications."
- "a set of programming instructions and standards for accessing a ~~Web-based~~ software application or Web tool. A software company releases its API to the public so that other software developers can design products that are powered by its service."
- "~~sets of requirements that govern how one application can talk to another~~."
- "An API specifies a software component in terms of its operations, their inputs and outputs and underlying types. Its main purpose is to define a set of functionalities that are independent of their respective implementation, allowing both definition and implementation to vary without compromising each other."

APIs can be messaged with in a variety of ways



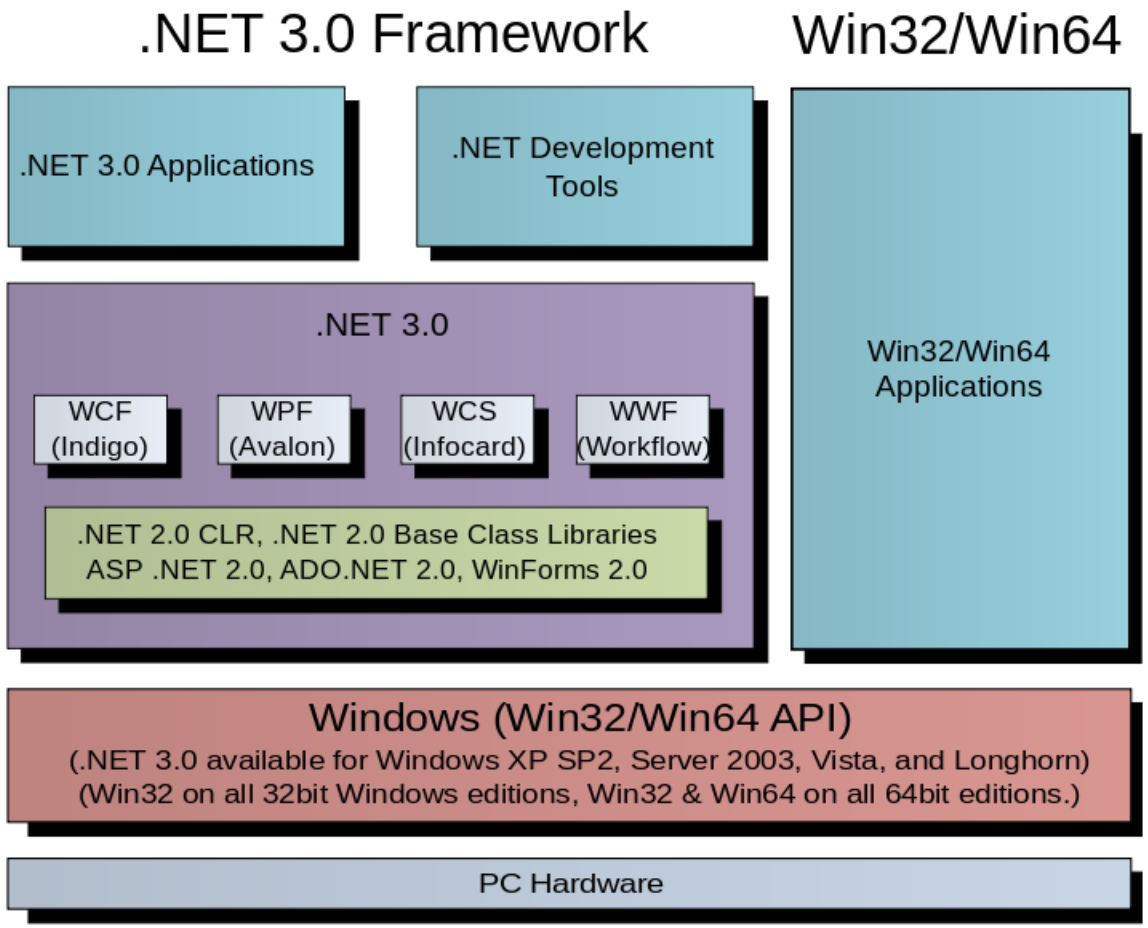
- APIs can "exist" locally
  - as in a simple program
  - communicate by function calls



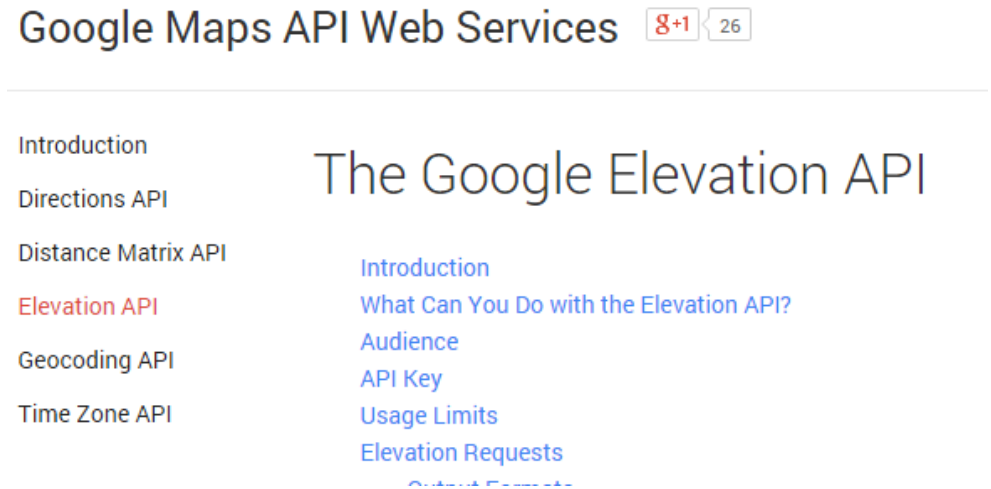
- Or they can run on a web server
  - to support web requests via XML or similar protocol

Windows has an API that documents function calls at a low level

- __WSAFDIsSet
- _IRDPSessionEvents
- accept
- AcceptEx
- AcquireSRWLockExclusive
- AcquireSRWLockShared
- ActivateAudioInterfaceAsync
- bind
- BindMoniker
- BSTR_UserFree
- BSTR_UserFree64
- BSTR_UserMarshal
- BSTR_UserMarshal64
- BSTR_UserSize
- BSTR_UserSize64
- BSTR_UserUnmarshal
- BSTR_UserUnmarshal64
- CloseCompressor
- CloseDecompressor
- CloseHandle
- closesocket
- CloseTrace
- CLSIDFromString
- …



Web-based APIs document what sort of web transactions will be handled by a given web service



- Google's Elevation API allows https requests for the elevation of a point
  - Low volume, ½¢ per call – but each account has a $200 monthly credit
- **`http://maps.googleapis.com/maps/api/elevation/json?locations=39.7391536,-104.9847034&key=`*`API_KEY`***

An API specification must contain several types of information

- Data Structures
- System Requirements
- Operations
- Interface

IBM i Access for Windows data transformation API list

Last Updated: 2021-08-31

The following IBM® i Access for Windows data transformation APIs are listed alphabetically.

> **Note:** IBM i Access for Windows data transformation APIs that accept strings are provided in Unicode versions. In these APIs, "ASCII" is replaced by "Wide" (for example, cwbDT_ASCII11ToBin4 has a Unicode version: cwbDT_Wide11ToBin4). These APIs are indicated in the table that follows. The Unicode versions have different syntax, parameters and return values than their ASCII counterparts.

- **cwbDT_ASCII11ToBin4**
  Use the IBM i Access for Windows cwbDT_ASCII11ToBin4 command.
- **cwbDT_ASCII6ToBin2**
  Use the IBM i Access for Windows cwbDT_ASCII6ToBin2 command.
- **cwbDT_ASCIIPackedToPacked**
  Use the IBM i Access for Windows cwbDT_ASCIIPackedToPacked command.
- **cwbDT_ASCIIToHex**
  Use the IBM i Access for Windows cwbDT_ASCIIToHex command.
- **cwbDT_ASCIIToPacked**
  Use the IBM i Access for Windows cwbDT_ASCIIToPacked command.
- **cwbDT_ASCIIToZoned**
  Use the IBM i Access for Windows cwbDT_ASCIIToZoned command.

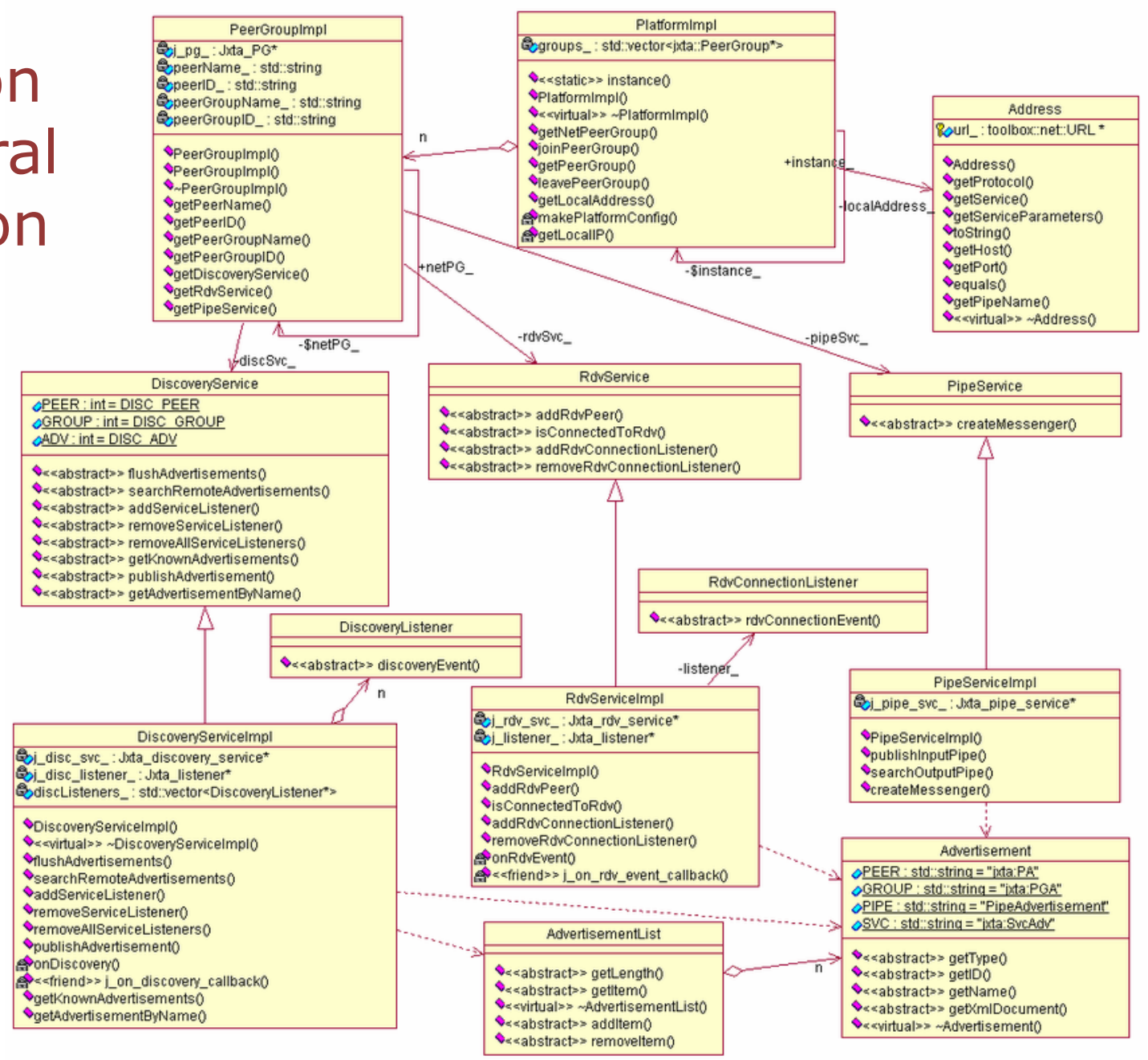
An API specification must contain several types of information

- Data Structures
- System Requirements
- Operations
- Interface

```
#ifndef FULLSPACETESTOPERATORS_H_
#define FULLSPACETESTOPERATORS_H_

#include "DOTk_Operators.H"

class FullSpaceTestOperators: public DOTk_Operators
{
public:
    FullSpaceTestOperators();
    virtual ~FullSpaceTestOperators(){}

    /* ========================================== Main Interface ========================================== */

    virtual Real Fval(const Vector& u, const Vector& z);
    virtual void Cval(const Vector& u, const Vector& z, Vector& cval);

    virtual void F_u(const Vector& u, const Vector& z, Vector &g);
    virtual void C_u(const Vector& u, const Vector& z, const Vector& dx, Vector& J_dx);
    virtual void adjC_u(const Vector& u, const Vector& z, const Vector& dy, Vector& J_dy);

    virtual void F_uu(const Vector& u, const Vector& z, const Vector& dx, Vector& H_dx);
    virtual void adjC_uu(const Vector& u, const Vector& z, const Vector& lambda, const Vector& du, Vector& adjC_du);

private:
    // unimplemented
    FullSpaceTestOperators(const FullSpaceTestOperators&);
    FullSpaceTestOperators operator=(const FullSpaceTestOperators&);
};

#endif /* FULLSPACETESTOPERATORS_H_ */
```

An API specification must contain several types of information

- Data Structures
- System Requirements
- Operations
- Interface



FRAMEWORKS

What is a framework? Is it different from an API?

- A framework is "a platform for developing software applications".
- Generally it will *contain* an API, but there is more to it:
  - A programming interface (API)
  - Classes of objects that interact significantly
  - An architecture
  - Typical interactions

When developing an application, designing and writing to a library is different than writing to a framework

VIRGINIA TECH
ece
BRADLEY DEPARTMENT OF ELECTRICAL & COMPUTER ENGINEERING

Using libraries:

- You design all of the system architecture
- Most system classes are left to you to design
- You code will use selected functionality from libraries but will be the "center"
- Emphasis is on laying out a complete solution, with the help of certain pieces of the solution coming from the libraries

Using frameworks:

- Much of the architecture is fixed
- Many or most classes are specified as part of the framework
- Your code will generally fit into certain spots in the framework
- Emphasis is on tailoring the framework to your specific needs

A *framework* has four characteristics that are different from a typical library

1. Inversion of control
2. Default behavior
3. Extensibility
4. Non-modifiable framework code

A *framework* has four characteristics that are different from a typical library

1. Inversion of control
   The app's flow of control is determined by the framework, not the code that you write
2. Default behavior

3. Extensibility

4. Non-modifiable framework code

A *framework* has four characteristics that are different from a typical library

1. Inversion of control
   The app's flow of control is determined by the framework, not the code that you write
2. Default behavior
   A framework has some "thing that it does" that is documented and can be worked with
3. Extensibility

4. Non-modifiable framework code

A *framework* has four characteristics that are different from a typical library

1. Inversion of control
   The app's flow of control is determined by the framework, not the code that you write
2. Default behavior
   A framework has some "thing that it does" that is documented and can be worked with
3. Extensibility
   It's possible to selectively override or add to certain behaviors of the framework

4. Non-modifiable framework code

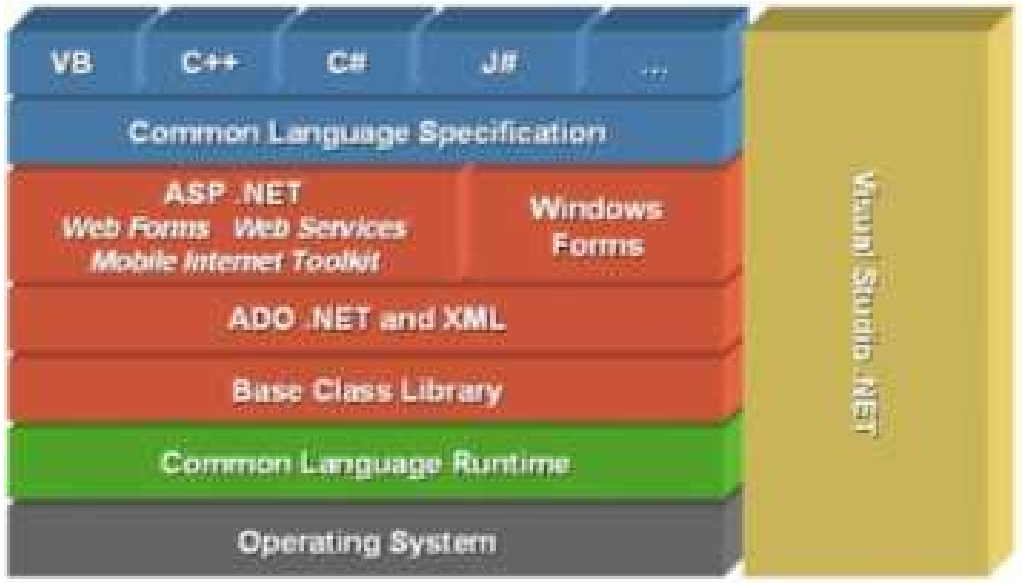
A *framework* has four characteristics that are different from a typical library

1. Inversion of control
   The app's flow of control is determined by the framework, not the code that you write
2. Default behavior
   A framework has some "thing that it does" that is documented and can be worked with
3. Extensibility
   It's possible to selectively override or add to certain behaviors of the framework

4. Non-modifiable framework code
   We modify its behavior through overrides or defined extension mechanisms, not by changing its code

As one example, .NET is a framework for creation of applications on modern versions of Windows

- Several APIs are included
- However, the way that a .NET Windows app will generally run is fixed
  - UI
  - Interaction with the system
  - Interaction with other apps



Some examples of popular frameworks

- .NET (Windows)
- Cocoa (Mac OS X)
- Cocoa Touch (iOS)
- Android Application Framework
- KDE Frameworks (Linux)
- Valve Source (gaming framework)
- Spring framework (Java)
- Sample code for a framework using DirectX

Programming to a framework is often less work but requires more research

1. Select the right framework (if any)
2. Understand and adopt the prevalent design
3. Code to the needed APIs
4. Look for large chunks of the problem that are already solved for you

Today's Objectives

Network Programming

- App-level network programming
- SSL/TLS
  - Public Key Infrastructure

Application Programming Interfaces

- What is an API
- How do we use an API

Frameworks

- What is a framework?
- Examples of common frameworks