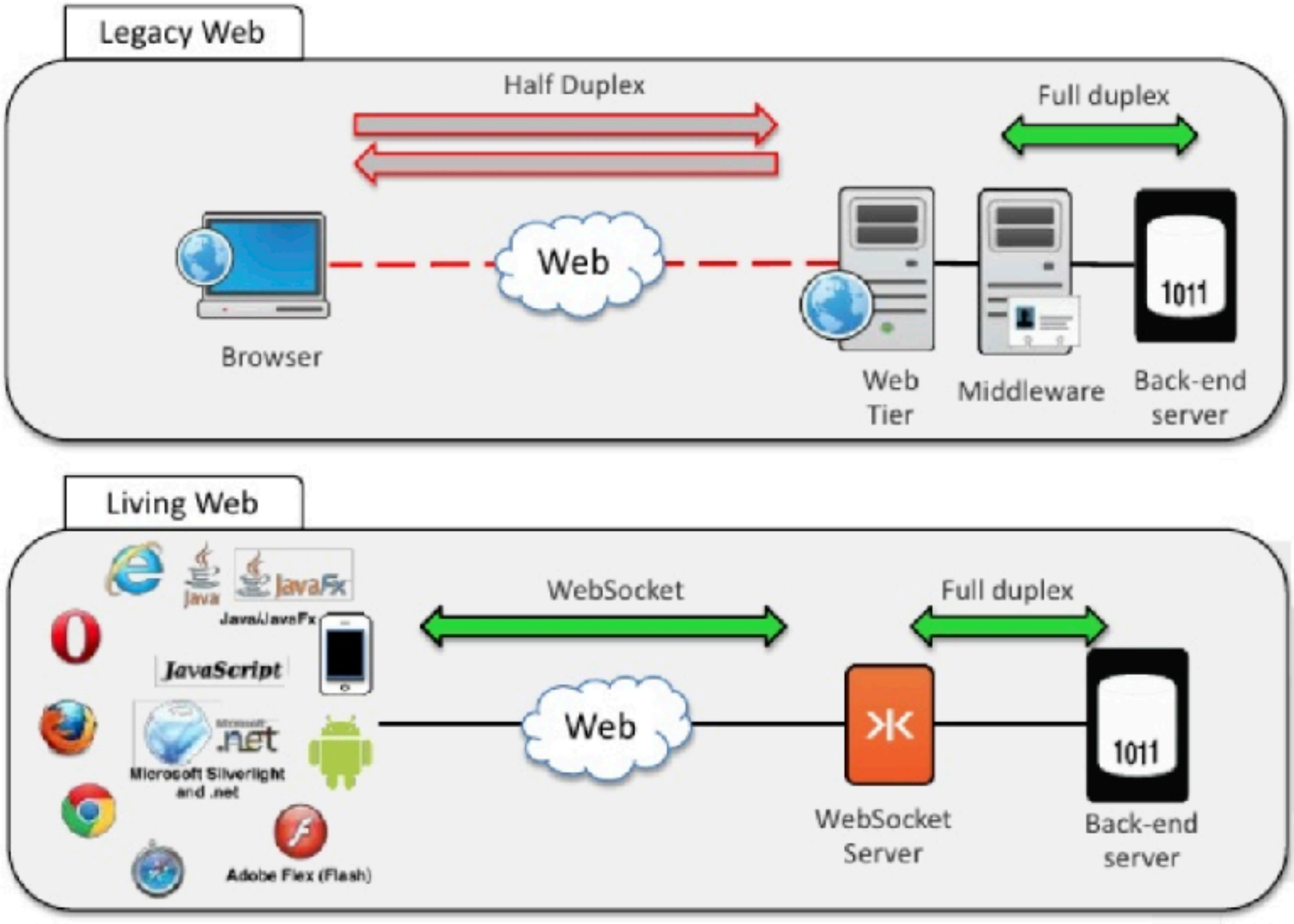
Web sockets

- Introduction to HTML5 WebSocket
- WebSocket API
- WebSocket Protocol

web architecture



http limitations

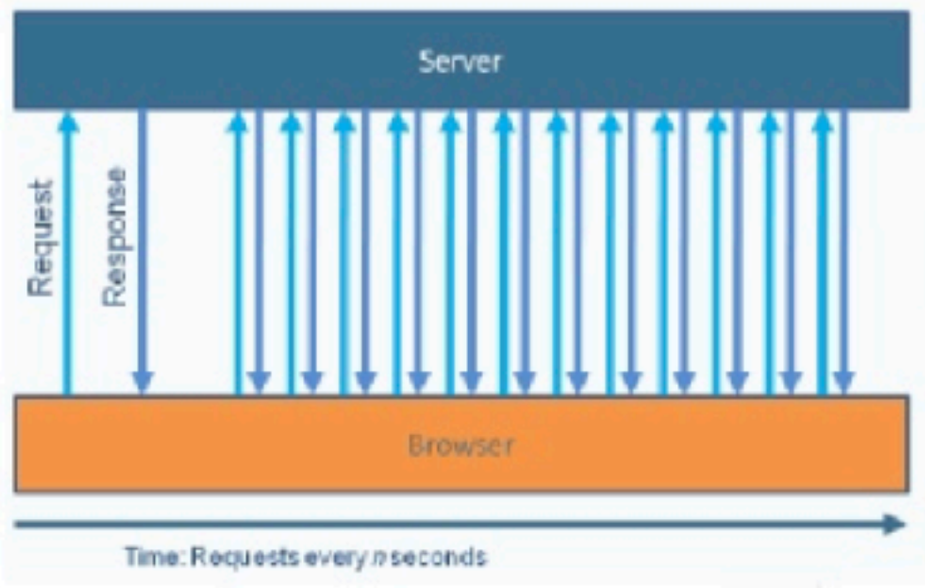
- Designed for document transfer
  - Request-response interaction
- Bi-directional but still half-duplex
  - Traffic flows in only one direction at a time
- Stateless
  - Header overhead

information is
sent with each HTTP
request and response

- AJAX (Asynchronous JavaScript + XML)
  - Content can change without loading the entire page
  - User-perceived low latency
- Comet
  - Technique for server push
  - Lack of a standard implementation
  - Comet adds lots of complexity

Polling

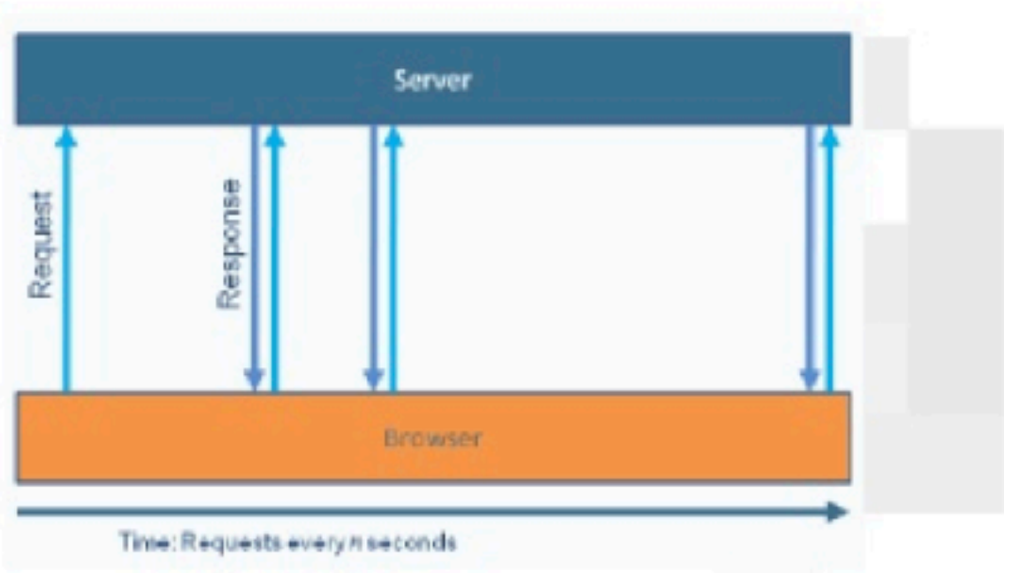
- Polling is "nearly real-time"
- Used in Ajax applications to simulate real-time communication
- Browser sends HTTP requests at regular intervals and immediately receives a response



long polling

a/k/a Asynchronous polling

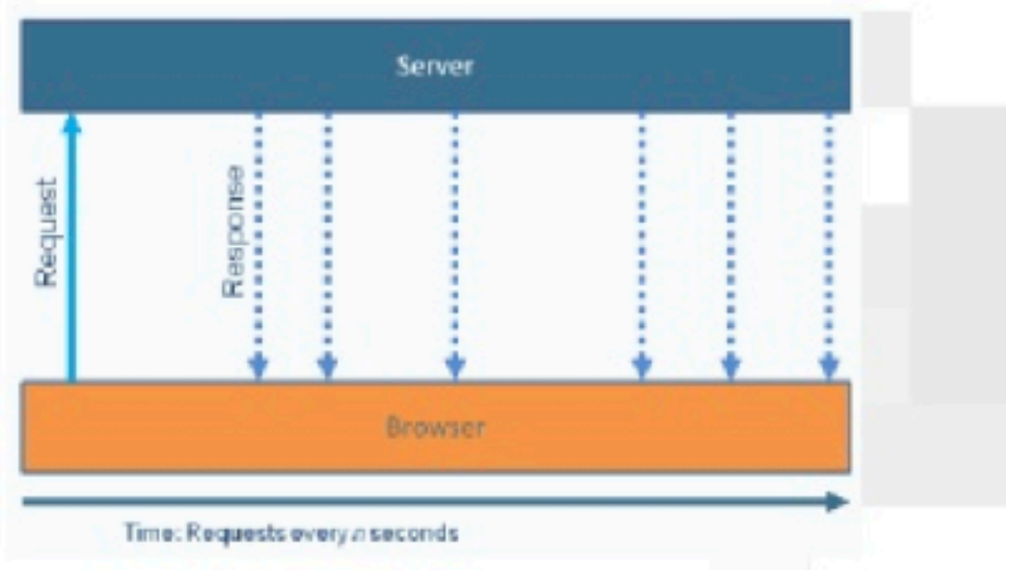
- Browser sends a request to the server, server keeps the request open for a set period
- Speed limited by response-request-response
- Request/response headers add overhead on the wire



streaming

- More efficient, but sometimes problematic
- Possible complications:
  - Proxies and firewalls
  - Response builds up and must be flushed periodically
  - Cross-domain issues

to do with browser connection limits



http request

Client

```
GET /PollingStock//PollingStock HTTP/1.1
Host: localhost:8080
User-Agent: Mozilla/5.0 (Windows; U; Windows NT 5.1; en-US;
rv:1.9.1.5) Gecko/20091102 Firefox/3.5.5
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: en-us
Accept-Encoding: gzip,deflate
Accept-Charset: ISO-8859-1,utf-8;q=0.7,*;q=0.7
Keep-Alive: 300
Connection: keep-alive
Referer: http://localhost:8080/PollingStock/
Cookie: showInheritedConstant=false;
showInheritedProtectedConstant=false; showInheritedProperty=false;
showInheritedProtectedProperty=false; showInheritedMethod=false;
showInheritedProtectedMethod=false; showInheritedEvent=false;
showInheritedStyle=false; showInheritedEffect=false;
```

http response headers

Server

```
HTTP/1.x 200 OK
X-Powered-By: Servlet/2.5
Server: Sun Java System Application Server 9.1_02
Content-Type: text/html;charset=UTF-8
Content-Length: 321
Date: Sat, 07 Nov 2009 00:32:46 GMT
```

- Total overhead: 871 bytes (example)
- Often 2K+ bytes
  - e.g. cookies

- W3C API and IETF Protocol
- Leverages Cross-Origin Resource Sharing
  - Enables web pages to communicate with a remote host
  - Shares port with existing HTTP content
  - Allows unlimited connections per Origin
    - Unlike HTTP which is limited by convention
    - One WebSocket handshake in progress per Origin
- Two schemes:
  - `ws://`
  - `wss://`

Checking for support

JavaScript

```
var status =  document.getElementById("support");
if (window.WebSocket) { // or Modernizr.websocket
  status.innerHTML = "HTML5 WebSocket is supported";
} else {
  status.innerHTML = "HTML5 WebSocket is not supported";
}
```

Web socket API

JavaScript

```
//Create new WebSocket
var mySocket = new WebSocket("ws://www.WebSocket.org");

// Associate listeners
mySocket.onopen = function(evt) {
};
mySocket.onclose = function(evt) {
  alert("closed w/ status: " + evt.code};
};
mySocket.onmessage = function(evt) {
  alert("Received message: " + evt.data);
};
mySocket.onerror = function(evt) {
  alert("Error);
};
```

Web socket API 2

JavaScript

```
// Sending data
mySocket.send("WebSocket Rocks!");

// Close WebSocket
mySocket.close();
```

	HTTP	WebSocket
Overhead	100s of bytes	2-6 bytes (typical)
Latency	New connection each time	None: Use existing connection
Latency (polling)	Wait for next interval	No waiting
Latency (long polling)	None, if request sent earlier + time to set up next request	No waiting