HTTP Protocol

- HTTP session
  - Sequence of network request-response transactions
- HTTP defines methods (verbs) that indicate the action to be performed on a specific resource
- Methods
  - **GET** → Requests a particular resource (e.g., a file)
  - **POST** → Submits data to be processed by a particular resource (e.g. inputting data into a database)
  - **HEAD** → Returns what GET returns, but without the actual resource (just headers)
  - **PUT** → Sends a representation of a particular resource
  - **DELETE** → Deletes the specified resource
  - TRACE → Echoes received request so client can check if any changes have been made
  - OPTIONS → HTTP method supported by the server for the specified URL
  - CONNECT → Converts request connection to TCP/IP tunnel
  - PATCH → To apply partial modifications to a resource
- HTTP servers must implement GET and HEAD and whenever possible OPTIONS

HTTP Protocol

- Response status codes (five classes)
  - 1xx Informational
  - 2xx Success
    - 200 → OK
  - 3xx Redirection
  - 4xx Client Error
    - 401→ Unauthorized
    - 404 → Not found
      - Useful “Page Not Found” error messages
        - http://www.evolt.org/node/4299
  - 5xx Server Error
- Full listing at:
  - http://en.wikipedia.org/wiki/List_of_HTTP_status_codes
- Example (using telnet)
  - http://www.cs.umd.edu/class/fall2005/cmsc433/HowWebServersWork.html

Web Services

- Web Service – Method of communication between two applications
- Web API (Application Programming Interface) that can be accessed over a network and executed at a remote system
    - Allows client applications to build interfaces to the service
- Web services share logic, data and processes through a programmatic interface across a network
- Two kinds
    - SOAP (Simple Object Access Protocol)
    - REST (Representational State Transfer)
- In General
    - REST → light-weight interactions
    - SOAP → secure, reliable interactions
    - Each has its advantages
- Example:
    - http://code.google.com/apis/maps/documentation/webservices/#WebServices

Web Services

- Services can range from simple requests to complicated business processes
  - Payment processing, content syndication
  - Currency conversion, language translation
- Any internet protocol can be used to build web services but HTTP and XML are often used
- By using web services your application can publish its functionality to the world
- Web services can be created in any programming language
- Web services allow data exchange between different applications and different platforms
  - With web services a company billing system can connect with a supplier server

Web Services (REST)

- REST (Representational State Transfer)
- An architectural style; not a protocol
- Allow different data formats (e.g., html, text, JSON)
- Advantages
  - Fast, language, and platform independent
- Can use SOAP web services as the implementation

Web Services (REST)

- Resources are represented by URLs
    - Resource → document, person, location
    - Each resource has a unique URL
        - Each resource does not need to have an actual page/document. It can be generated dynamically
    - A resource is considered a "noun"
  - Operations are performed via HTTP methods (GET, POST, PUT, DELETE)
    - Methods are considered "verbs"
- REST → **designed to operate with resource-oriented services (locate/manipulate resource)**

Web Services (REST)

- **Example:**
  - Web service that allows individuals to manage file backups
  - Each backup has an URL http://backupFake.doesnotexist.org/backups/1938
  - Using HTTP GET we can get the backup
  - Using HTTP PUT we can update a backup
  - Using HTTP POST we can upload a backup
    - We can receive a URL that corresponds to the new backup
  - Using HTTP DELETE we can delete a backup
- Notice that REST relies on a familiar approach (HTTP methods) to ask for services (we don't need to create a new interface/approach)

Web Services(SOAP)

- SOAP (Simple Object Access Protocol)
- XML-based protocol for accessing web services
- Platform and language independent
- Designed as a way to package remote procedure calls into XML wrappers
- Disadvantages
  - Slow – Uses XML format that must be parsed to be read
  - Consumes more bandwidth
- SOAP request
  - XML document
  - Has three components
    - Envelop → defines document as SOAP request
    - Body → provides information about the call and responses
    - Optional header and fault elements
- SOAP response is an XML document
- SOAP cannot use REST because it is a protocol
- SOAP defines standards to be strictly followed

Web Services (Platform Elements)

- WSDL (Web Services Description Language)
  - XML-based language for describing and locating web services
  - W3C standard
  - Similar to a contract that defines the interface that services offers
  - It is machine-readable
- UDDI (Universal Description, Discovery and Integration)
  - Directory service where companies can search and register for web services described by WSDL

REST vs. SOAP

- https://www.javatpoint.com/soap-vs-rest-web-services
- SOAP is a procotol; REST architectural style
- SOAP can’t use REST as it is a protocol; REST can use SOAP web services
- SOAP uses services interface to expose logic; REST uses URIs
- SOAP defines standards to be strictly followed; REST does not define too many standards
- SOAP requires more bandwidth and resources; REST requires less
- SOAP defines its own security; REST inherits security from underlying transport
- SOAP permits only XML data; REST permits different data format

References

- http://en.wikipedia.org/wiki/Hypertext_Transfer_Protocol
- https://www.javatpoint.com/soap-web-services

Promises

- Promise - object that represents the eventual completion (or failure) of an asynchronous operation.
  - We attach callbacks to the promise object
  - Allows promise chaining
    - Execution of two or more asynchronous operations back to back where results of one step are used by the next
- **Example:** PromisesBasics.html, PromisesFib.js
- Reference
  - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Using_promises

Modules

- Immediately Invoked Function Expression
- **Example:** ImmediatelyInvokedFunctionExpression.html
- **Example:** ModuleImplementationViaIIFE.html
- Modules in Node
  - Define your functions a file
  - To export add the function to the exports object
  - **Example:** https://nodejs.org/docs/v0.5.0/api/modules.html

Modules

- Two module specifications:
  - CommonJS and AMD (Asynchronous Module Definition)
- AMD
    - Designed with the browser in mind
    - Popular implementation – RequiredJS
- CommonJS
  - For general-purpose JavaScript (e.g., NodeJS)