1. Interfacing with webservice APIs
   a. Common webservice technologies:
      i. SOAP: Highly structured, well-suited for complex transactions and datasets. Not really “simple.” Part of W3C specification. Security framework is part of the W3C SOAP standard. Most SOAP webservice APIs have a similar feel.
      ii. REST: Well-suited for sequential inter-dependent transactions. Security framework is part of the individual webservice implementation. Programming details vary from one REST service to the next.
   b. Handling access security & authentication needs
      i. Developer IDs and keys – controls access to the API itself
      ii. Username/password authentication – controls access to the data
      iii. Auth/oAuth authentication – controls access to the data, is less dependent on knowing when a user changes his password
   c. Parsing and deserializing results
      i. XML – Good for representing complex datasets with a variety of datatypes, as well as sparse datasets. Used for both sending data (SOAP) and receiving data (SOAP & REST)
      ii. JSON – Easier and faster to parse than XML, sometimes has difficulty with null/zero/blank fields. Most commonly used for receiving data.
      iii. Plain Text – Unstructured datasets. Used for receiving data that is meant for displaying to a user (HTML etc)
      iv. Delimited Data – Usually used for reporting. Used for receiving data in a format such as CSV for importing into other applications (databases, spreadsheets, etc).
   d. Programming tools
      i. Libraries and extensions for most languages (php, .NET, Java, etc.) handle the low-level tasks of building data structures, making HTTP calls, data validation and serialization/deserialization.
      ii. Some languages provide XML and JSON parsing functions and services as part of the base language, or use “under the hood” facilities such as Apache libraries that are transparent to the programmer.
   e. Debugging tools
      i. SOAPUI – decodes SOAP webservice WSDLs and provides ability to mock-up a webservice request by hand. Can record SOAP traffic
      ii. SOAPUI – Also works with REST webservice’s resources and methods, sets parameters, creates mock-ups
      iii. Internet Explorer – easy way to examine XML files and their structure

2. What common (non-genealogy) webservice APIs are available?
   a. Translation – GoogleTranslate
   b. Mapping – GoogleMaps
   c. Geographic places & names - GNIS
   d. Charting
3. FamilySearch.org API
   a. Standard “REST” architecture
   b. Variety of functions/methods and of data that can be searched on or retrieved.
   c. People, Places & Names, Families, Cemeteries, Citations, Assertions
   d. Lookups, matchings, updates/creation
   e. Devnet.familysearch.org/docs

4. Bringing them together
   a. Common obstacles:
      i. Datatype mapping – especially dates and places
      ii. Multiple disparate authentication systems
   b. Mapping – GoogleMaps, Google Places
      i. Show relevant places for a person or group
      ii. Show migration paths
   c. Translation – GoogleTranslate
      i. Help in understanding transcribed text
   d. GNIS (USGS government service)
   e. CENSUS API
   f. GeoNames.org

5. Resources, where to go for more information
   a. www.w3schools.com/soap/soap_intro.asp
   b. “RESTful Web Services: Web services for the real world” (Richardson & Ruby, O’Reilly 2008, 978-0-596-52926-0)
   c. “REST in Practice” (Webber et al, O’Reilly 2010, 978-0-596-80582-9)
   d. “Programming Web Services with SOAP” (Snell, O’Reilly 2001, 978-0-596-00095-0) (dated)
   e. Lots of online articles and tutorials – Google/Bing search