Dynamically constructing UIs, Implicit Intents, REST, and CURL
Client / Server Architecture

- Server maintains durable state
- Clients connect to server to access/modify state
  - Supports multiple clients simultaneously

The Client makes request for service to server.

The server responds to that request.
Server state typically kept as “tables”

- Managed by databases
  - E.g., Movies table at themoviedb.org

<table>
<thead>
<tr>
<th>id</th>
<th>title</th>
<th>original_title</th>
<th>original_language</th>
<th>popularity</th>
<th>vote_count</th>
<th>vote_average</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- We won’t do server-side stuff in CS 126.
Application Programming Interfaces (API)

- APIs are the interface between the client and server
- Like everything, there are good & bad APIs
- Some best practices to be aware of
RESTful APIs

- REST = Representational State Transfer
- Stateless, client-server, cacheable communications protocol
- Generally built on HTTP/HTTPS
- Supports Creating, Reading, Updating & Deleting (CRUD)
- Language/Platform independent
- REST operations should be self-contained

Nice overview:
- https://twincl.com/programming/*6af/rest-api-design
REST Concepts: Verbs

- **Standard Verbs:**
  - GET: Read an existing resource (IDEMPOTENT, SAFE)
  - POST: Create a new resource ()
  - PUT: Update an existing resource (IDEMPOTENT)
  - DELETE: Delete an existing resource (IDEMPOTENT)

- **Other Standard Verbs:**
  - OPTIONS: Get list of what verbs are allowed (IDEM, SAFE)
  - HEAD: Get headers (e.g., metadata) of GET (IDEM, SAFE)
  - PATCH: Update part of existing resource ()

- You can create your own, but some discourage it
REST Concepts: Nouns (a.k.a. resources)

- Mostly Describe Resources
  - /movies, /movies/17, /movies/17/reviews
  - Prefer plurals, use numbers to index into collection
  - Narrow selection through use of query string
    - /movies?genre=13&year=2015

- Can also specify utility APIs
  - /search?q=keyword
To Do for Thursday

- Watch Lesson 3 of the Google/Udacity class
  - Create new activities and navigate Apps with Intents

- Make detail view for Election Predictions app (for next week)
  - Click on a state, get a detail page about that state, including
    - All candidates in order expected percentage of the vote, listing
      - Name, party (letter abbreviation), and projected fraction of the vote

- Start thinking about a final Android project
  - Something interactive (i.e., two people could interact through the app).