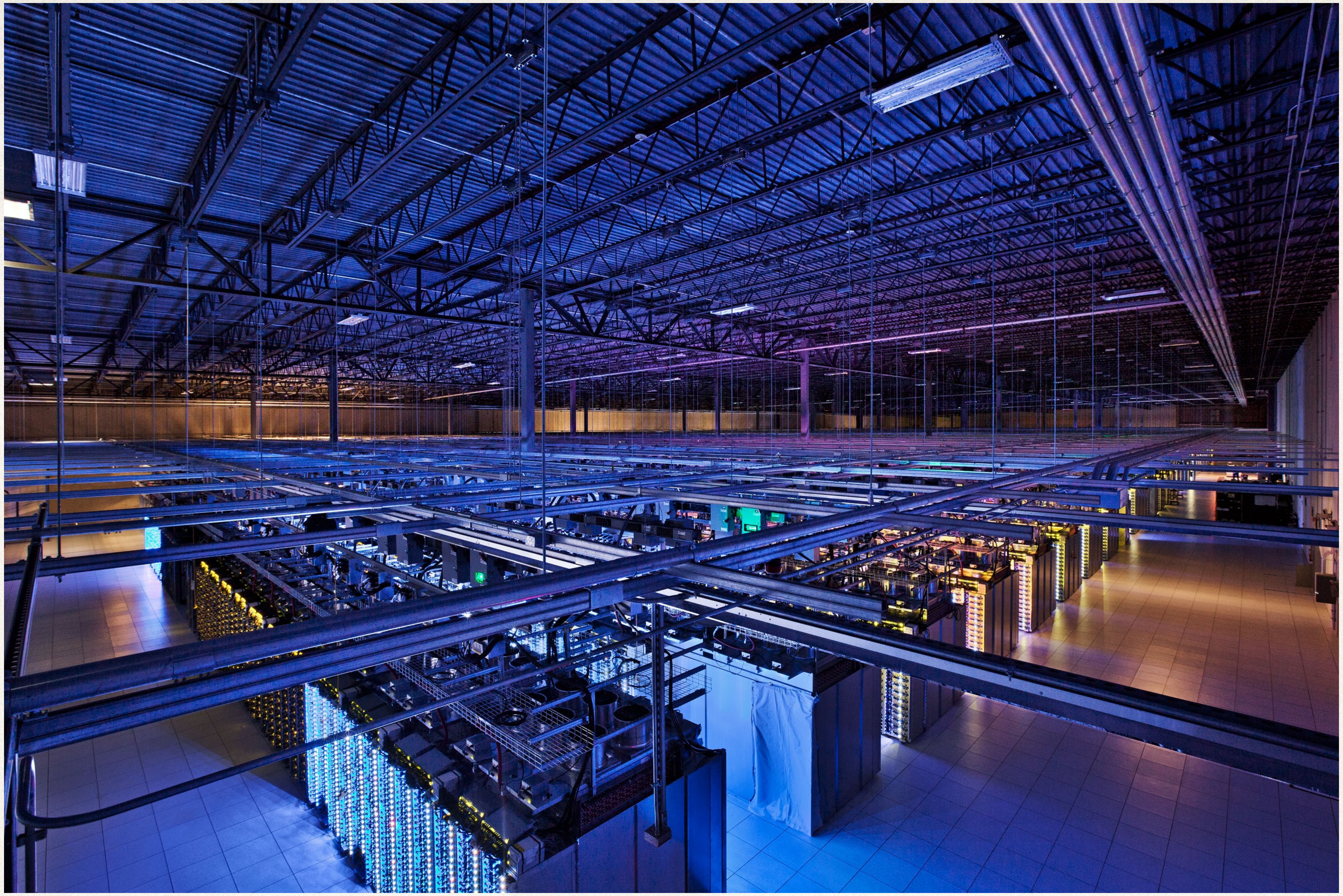


Backends and Databases

Dr. Sarah Abraham

University of Texas at Austin
CS329e
Spring 2020

What is a Backend?

- Server and database external to the mobile device
- Located on remote servers set up by developers
- Provides app information to users
- Allows for:
  - Reuse of data across user-base
  - Security controlled by the developers
  - More reliable storage

When to Offload?

- When should you offload to the backend?
  - Information is regularly updated
  - Systems are regularly updated
  - Calculations require better hardware
  - User data collected for analytics etc
- “Cloud” model is extremely popular right now

Network

- Computers communicate via sockets (ports)
- Port provides connection into computer
- Thousands of ports available on a computer
- Ports can be turned off for security
- Predefined ports have specific usages
  - Port 80 (http/insecure)
  - Port 443 (https/secure)

TCP/IP

- Transfer Control Protocol/Internet Protocol)
- Dominant network protocol
- Two addressing protocols, IPv4 and IPv6
- App must be able to talk to both networks:
  - `NSURLConnection`
  - `URLSession`

Networking Frameworks

- `NSURLConnection` is older framework
- `URLSession` replaces `NSURLConnection`
    - Recommended networking framework
- `AFNetworking` / `AlamoFire` are third-party frameworks
    - Popular and stable
    - Built on top of `Connection` and `Session`
    - `AlamoFire` is Swift version

URLSession

- Provides API for downloading/uploading content over the Internet
- Built-in support for authentication and execution of background tasks
- Supports data, file, ftp, http and https URL schemes
- Support for proxy servers
- Allows canceling, restarting, resuming and suspending tasks and downloads

Configuration Objects

- `URLSessionConfiguration` defines behaviors and policies for uploading and downloading
    - *Shared* handles basic requests
    - *Default* allows for incremental data transfer
    - *Ephemeral* does not write caches, cookies, or credentials to persistent store
    - *Background* allows download/upload as a background task

Choosing a Configuration

- *Shared* has limited customization but allows for easy URL fetching
    - Does not create a configuration object
    - Accesses the property directly
- *Default* is similar to *shared* before customizing but can obtain data incrementally
    - Creates default configuration object
    - Stores credentials in user's keychain
- *Ephemeral* allows for private sessions
    - Stores all data to RAM
    - Only writes to disk when told to write contents to file
- *Background* sessions can run even when the app is off or suspended
    - `isDiscretionary` allow session to optimize for performance

Additional Configurations

- Timeout values
- Caching policies
- Security policies
- Background transfers
- HTTP and proxy policies

Session Tasks

- `URLSessionTask` performs actual work for retrieving data
- *Data tasks* send and receive data using NSData objects
  - Used for short requests and small amounts of data
  - Uses HTTP GET
- *Upload tasks* send and receive large amount of data
  - Uses HTTP POST and PUT
- *Download tasks* retrieve data in the form of a file

Session Delegates

- A session's *delegate* tracks when events occur:
  - Authentication requests from server
  - Data arriving from server
  - Any failures in session
- If no event-tracking features are required, can pass in delegate as nil

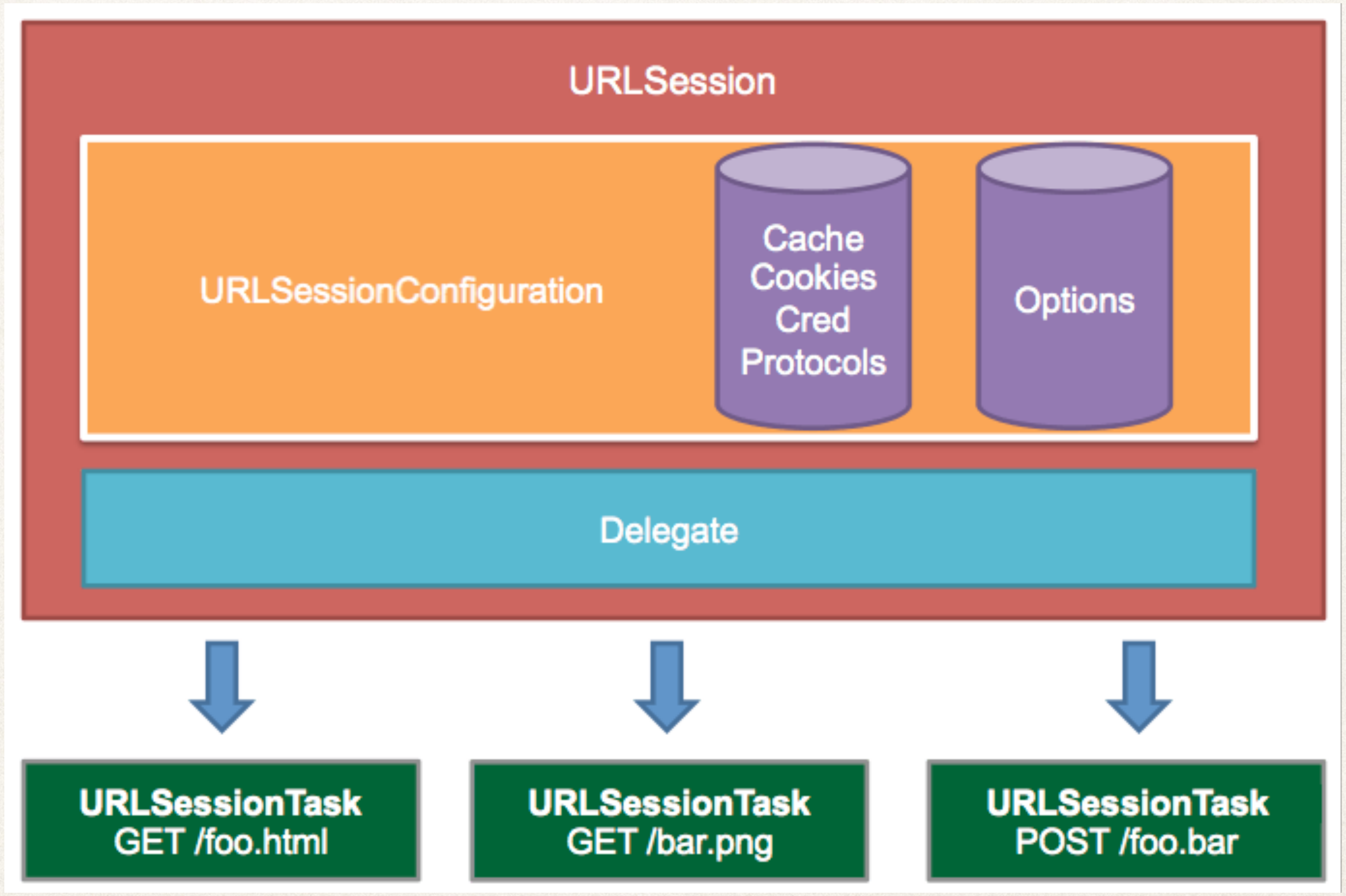
Custom Protocol

- Implement basic delegate functionality via a protocol
- Handles errors and return data:

```
protocol DataProtocol {
   func responseDataHandler(data:[return type])
   func responseError(message:String)
}
```

- Associated with view controller (like any other protocol)

Session Overview



(www.raywenderlich.com)

Session Summary

- `URLSession`: high-level session object
- `URLSessionTask`: object that contains one or more task objects
- `URLSessionDataTask`: subclass of sessionTask for direct data retrieval
    - `dataTask(with: URLRequest)`
    - `dataTask(with: URL)`
- Tasks begin in suspended state
- Start task by calling resume

Instapoll: URLSession

- What is the basic URLSessionConfiguration?
  - *Default*
  - *Ephemeral*
  - *Background*
  - *Shared*

URLSessionDemo

Network Payloads

- Data to be sent across the network
- Structured to be read on both ends:
    - JSON (JavaScript Object Notation)
    - XML (eXtensible Markup Language)

XML

- Tags defined by angle brackets
- Content placed within tags
- Tags can be nested

```
<element>
 <item>First item</item>
 <item>Second item</item>
 </element>
```

- Nested elements are children of the enclosing elements

XML Example

```
<?xml version="1.0" encoding="UTF-8" ?>
<data>
    <current_condition>
        <cloudcover>16</cloudcover>
        <humidity>59</humidity>
        <observation_time>09:09 PM</observation_time>
    </current_condition>
</data>
```

JSON

- Model for objects and arrays
- Presents hierarchical structures (like XML)
- Easier to structure and parse
- Objects are unordered name/value pairs: `{name1: value1, name2: value2}`
- Arrays are order collections of values: `[value1, value 2]`
- Objects and arrays can be values

JSON Example

```
{
    "data": {
        "current_condition": [
            {
                "cloudcover": "16",
                "humidity": "59",
                "observation_time": "09:09 PM",
            }
        ]
    }
}
```

JSON Encoding/Decoding

- Networks send streams of bytes
  - Data must be encoded (serialized) to send
  - Data must be decoded (deserialized) to receive
- `JSONSerialization` provides functionality for converting JSON data to dictionaries, arrays, numbers, or Strings
  - Must use try/catch to check if JSON data is valid
  - Must use optional unwraps to ensure values exist

Encodable/Decodable

- Protocols for encoding and decoding between representations
- Many data representations already encodable/decodable
  - Custom types must conform to `Encodable`/`Decodable` protocols to be codable
- Can encode/decode to XML, plists, JSON, etc
- Allows for more efficient handling of networked data

App Transport Security

- Handles security between app and web
- Required for app using `NSURLConnection` or `URLSession`
- Will reject insecure connections
- Exceptions can be added in Info.plist
  - App Transport Security Settings
  - Be careful how you do this!
- <http://www.neglectedpotential.com/2015/06/working-with-apples-application-transport-security/>