Web Design and Development

Gerald Aden

Week 1

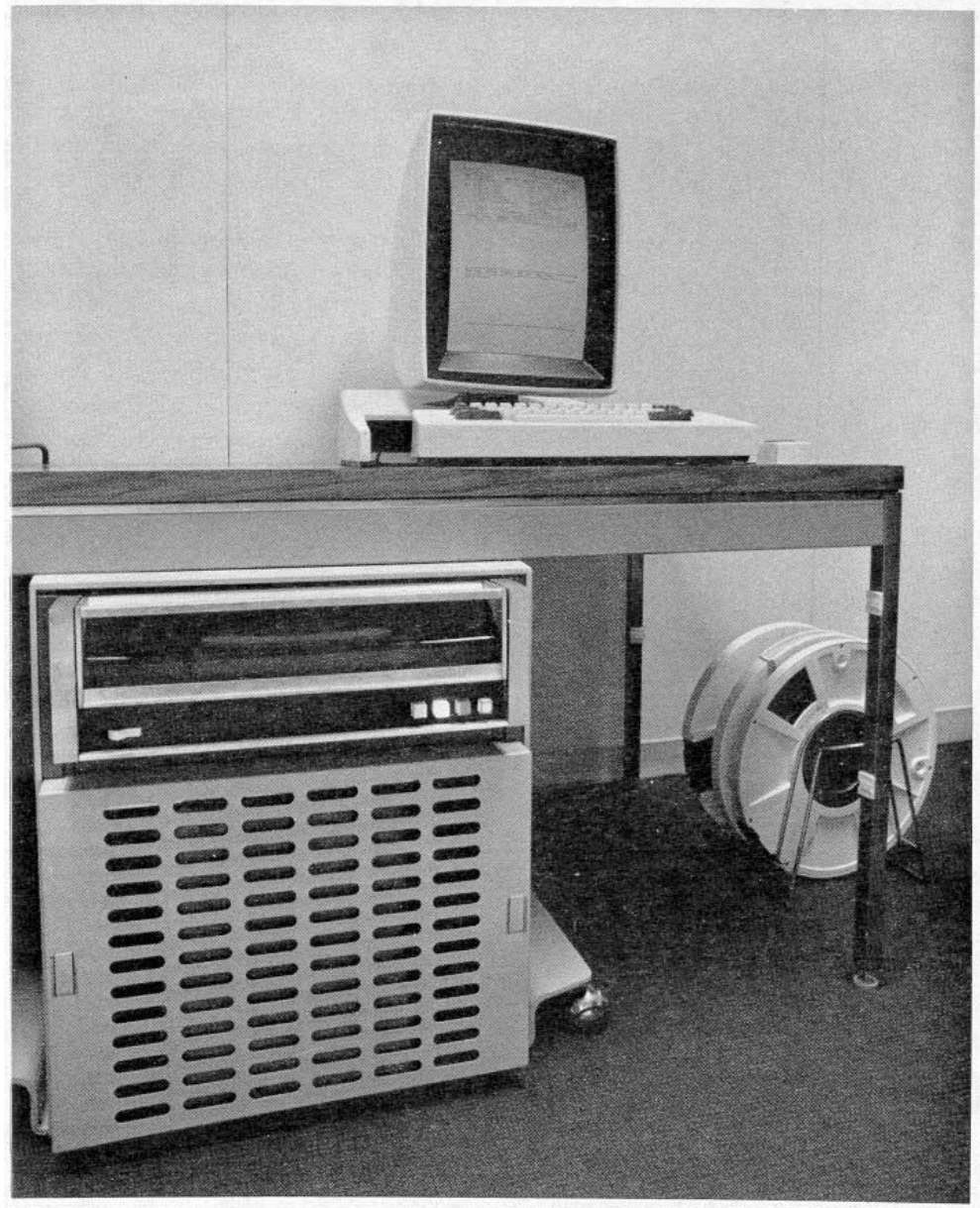
Introduction

Introduction

Introductions: Me

- BS in Math/Computer Science
- Over 30 years experience in software development
  - Xerox, aerospace, finance, telecommunications, legal, banking, transportation, healthcare
- Adjunct faculty at PCC teaching various software development courses
- Currently work for Surescripts
- Current technical interests: IoT, Linux and Containers
- 1 year at OIT
  - gerald.aden@oit.edu


this shall not be forgotten. Thanks internet!
1969-10-29 22:30
IN COMMEMORATION
of the first ARPANET
message, sent to the
Stanford Research
Institute from 3420
Boelter Hall.
It marks the most
evident origin of
today's internet.



Xerox

- Ethernet
- Network File System
- Mouse
- Graphical Editor
- Interpress/Postscript/PDF
- OOP (Smalltalk)
- Desktop interface with windows
- Laser Printer
- MVC (Smalltalk)

Introduction
Introductions: You

- Name
- Major and year
- Experience:
  - Web development
  - C#
  - GIT
  - Relational databases
  - Visual Studio
- Technical interests

Introduction
Course: Description

- Web application development
- Focus on development/engineering rather than design
- Broad rather then deep (up to you)
- Goal is to become acquainted with important aspects of Web application development in a real world environment

Introduction
Course: Sessions

- Lecture: Monday 6 - 7:50
  - Quiz and lecture
  - Break
  - Demos
- Lab: Saturday 11 – 1:50
  - Exercises
  - Work on class project
  - Explore and get help
  - Presentations (extra credit)

Introduction
Course: Communication

- Email
- Blackboard
- GitHub

Introduction
Course: Topics

- Backend development (Microsoft ASP.NET MVC)
- Database/Object Relational Mapping (ORM)
- Dependency Injection
- Front End Development (HTML, JavaScript, CSS, JQuery or Vue)
- Debugging/Diagnostics
- Testing (Unit and Acceptance)
- Security
- REST
- Continuous Integration (CI)
- Deployment

Introduction
Course: Grading

- Attendance/Participation
- Lab Exercises 20%
- Quizzes (open notes) 10%
- Final (open notes) 20%
- Project 50%
- Extra Credit (presentations)

Introduction
Course: Project

- Incorporate what you learn in class
  - Backend development using ASP.MVC
  - Database with Object Relational Mapping
  - Front end user interface using HTML/CSS/JavaScript
  - Dependency Injection
  - REST
  - Unit Tests
  - Continuous integration
  - Deployment (extra credit)

Introduction
Course: Tools

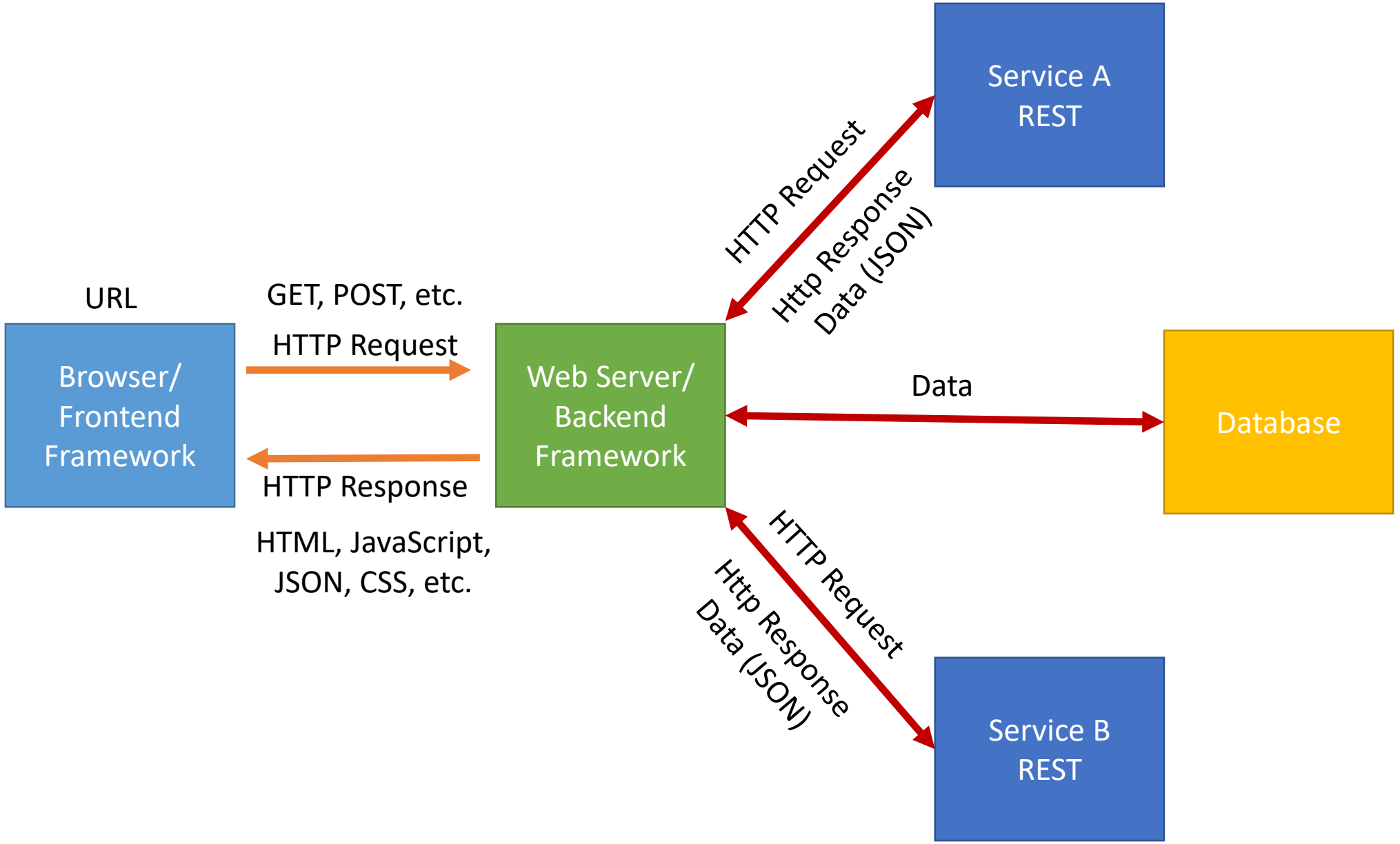
- Visual Studio
- C#
- Git/GitHub
  - Command Line, SourceTree
- Fiddler
- Postman
- Chrome Developer Tools

Introduction
Course: Resources

- https://www.syncfusion.com/resources/techportal/ebooks
- http://www.asp.net/aspnet
- Others listed in Blackboard throughout the course

HTTP

- Hypertext Transport Protocol
- Industry standard developed by the World Wide Web (WWW) consortium in conjunction with industry (Google, Microsoft, etc.)
- Specifies content of data payload exchanged between clients and servers
- Resources identified using URLs
- Actions specified by methods (GET, POST, DELETE, etc.)
- Important to understand the basics


Service A
REST
HTTP Request
Http Response
Data (JSON)
URL
GET, POST, etc.
HTTP Request
Browser/
Frontend
Framework
Web Server/
Backend
Framework
Data
Database
HTTP Response
HTML, JavaScript,
JSON, CSS, etc.
HTTP Request
Http Response
Data (JSON)
Service B
REST

Web Development Frameworks Backend

- **Microsoft ASP.NET (.NET)**
- Node (JavaScript)
- Rails, Sinatra (Ruby)
- Nancy (.NET)
- Microsoft ASP.NET (Core)
- Django, Flask, Others (Python)
- Spring, Others (Java)

Web Development Frameworks Frontend

- **JQuery**
- Angular
- React
- Ember
- **Vue**
- Others

ASP.NET MVC

- Model View Controller
  - Rails, Java Spring, Node Sails
- Architecture pattern
- Standards based
- Lowered the level of abstraction from Web Forms
- Separation of concerns
- Testability
- Flexible/Customizable
- Last version is 5

ASP.NET MVC
Models: What are they?

- In memory representation of:
  - Concrete concepts in your application domain
    - student, patient, friend etc.
  - Abstract concepts in your application domain
    - login attempt, temperature reading, etc.
- Content (properties) comprised of data required by your application
- Generally only contain data (no methods)
- Populated by controllers from data they receive
  - Sent by browser
  - Retrieved from databases and/or services (REST)
- Just a C# class

ASP.NET MVC
Models: Entity Models

- Contains data retrieved from a database
- Used as source of data for view models

ASP.NET MVC
Models: View Models

- Contains data from one or more entity models
- Used by MVC views as a source of data

ASP.NET MVC
Controllers

- Provide an HTTP endpoint for clients (browsers)
- Class in C#
- Contain **actions** (class methods) that implement an interface to some piece of application logic
- Requests sent over HTTP from a client are **routed** to a controller action
  - Not a file as in traditional Web applications/sites
- Process requests (anything)
- Returns response (HTML, JSON, file, etc.) back to client

ASP.NET MVC
Controllers

- Should be lightweight (not much if any application logic)
- Best practice is to utilize other classes to perform application logic (access data, call REST services, calculations, etc.) to facilitate testability
- Name is typically [DomainEntity]Controller
- Create and populate models using data they receive from databases and/or services (REST)

ASP.NET
Controllers: Routes

- Control what controller action a URL is “mapped” to
- Can also specify data mapped from values that are part of the URL
- Routes are configured by the developer
- URLs
  - http://www.mysite.com/Details.aspx?personId=123
  - http://www.mysite.com/person/details/123

ASP.NET MVC
Views

- Templates
- Handle Web page rendering
- Any data to be inserted in a Web page is in a defined model
- Invoked by a controller
- Contain mostly HTML
- Contain display logic (careful not to include application logic)
- Razor syntax (C#)
  - Conditionals
  - Loops
  - Placeholders for data

ASP.NET MVC

Demo: Creating a Project

- Creating an ASP.NET MVC project
- Folder structure
- Conventions (actions -> views, etc.)
- Support files
- Anatomy of a controller
- Route basics
- Show app (responsive by resizing browser, etc.)