

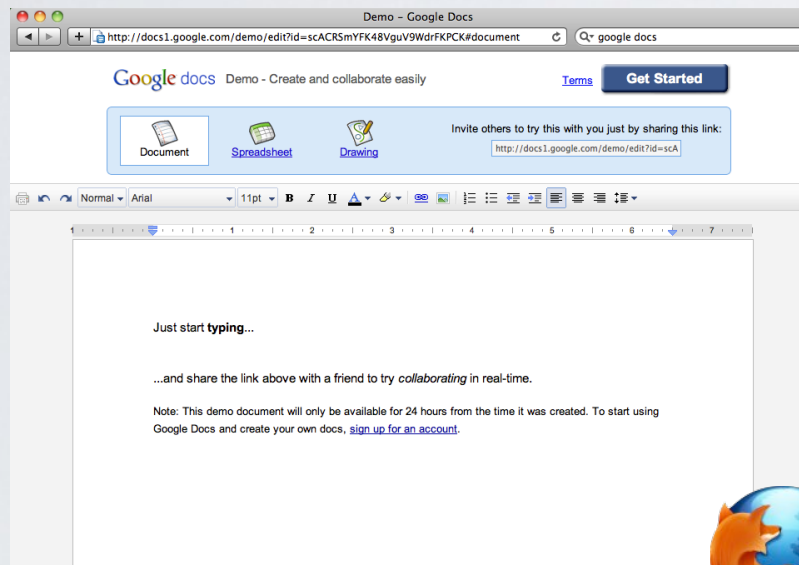
The HTTP protocol

Thierry Sans

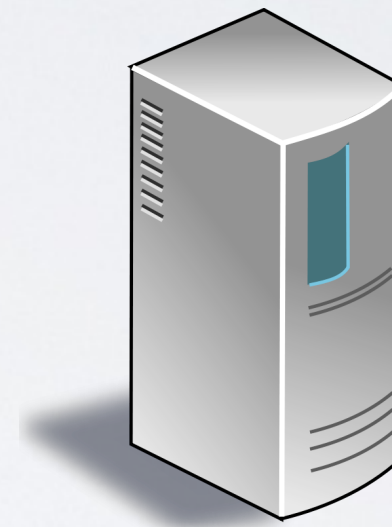
Anatomy of a Web Application

Client Side

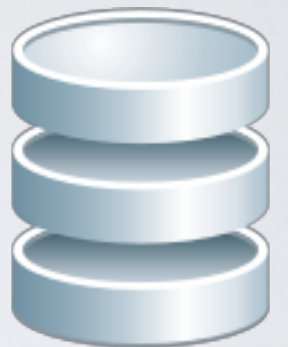
Server Side



Web Browser



Web Server



Database

The HTTP protocol

Network protocol for requesting/receiving data on the Web

- Standard TCP protocol on **port 80** (by default)
- **URI/URL** specifies what resource is being accessed
- The **request method** specified with a command

Let's look at what a web server does

telnet to a web server

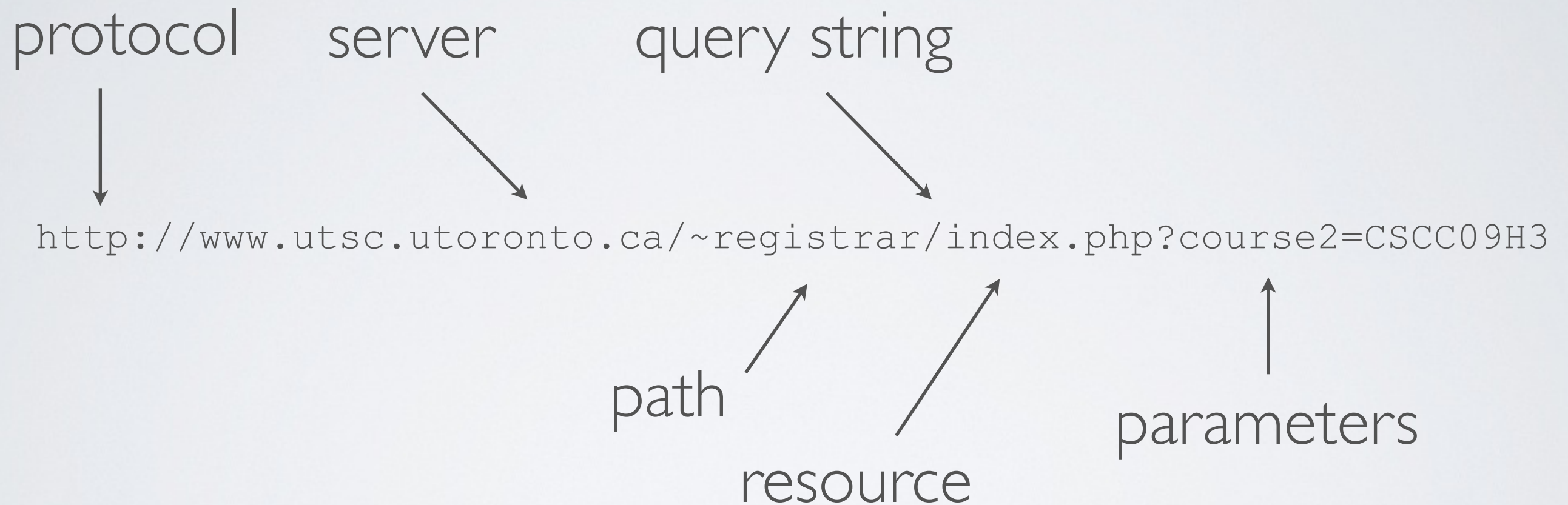


```
> telnet www.utsc.utoronto.ca 80  
GET /
```



enter HTTP requests

Anatomy of a URL



HTTP Request Methods

- **POST** - add an unidentified resource
- **PUT** - add a an identified resource
- **GET** - get a resource
- **PATCH** - update a resource
- **DELETE** - delete a resource
- and others HEAD, TRACE, CONNECT, OPTIONS

HTTP Request

- **Method** - POST, PUT, GET, PATCH, DELETE ...
- **Query String**
- **Headers** - key/value pairs
- [optional] **Body** - data

Using the command **curl**

```
$ curl options url
```

```
-v verbose
```

```
--request request_method
```

```
--data request_body
```

```
--header header
```


HTTP response

- **Status code**
- **Headers** - key/value pairs
- [optional] **Body** - data

HTTP response status codes

- 1xx - information
- 2xx - success
- 3xx - redirection
- 4xx - client error
- 5xx - server errors

Method properties

An HTTP request/response

- may have a request body
 - may have a response body
 - may not have side effects a.k.a safe
 - may have the same result when called multiple times a.k.a idempotent
- ➡ the choice is left to the programmer

What the standard recommends

Method	Request Body	Response Body	Safe	Idempotent
POST				
PUT				
GET				
PATCH				
DELETE				