Theme 10-1

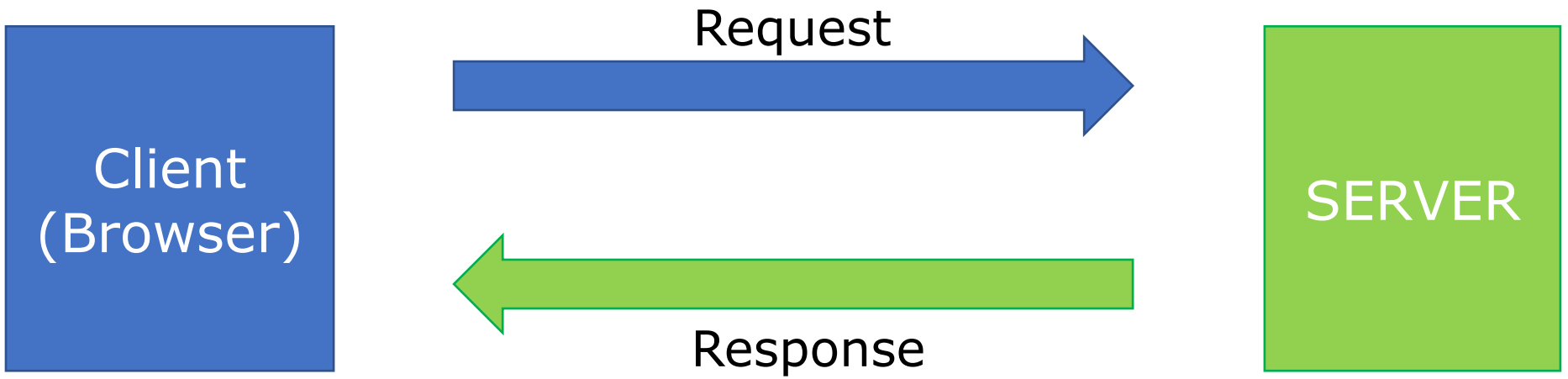
RESTful API

What is RESTful APIs?

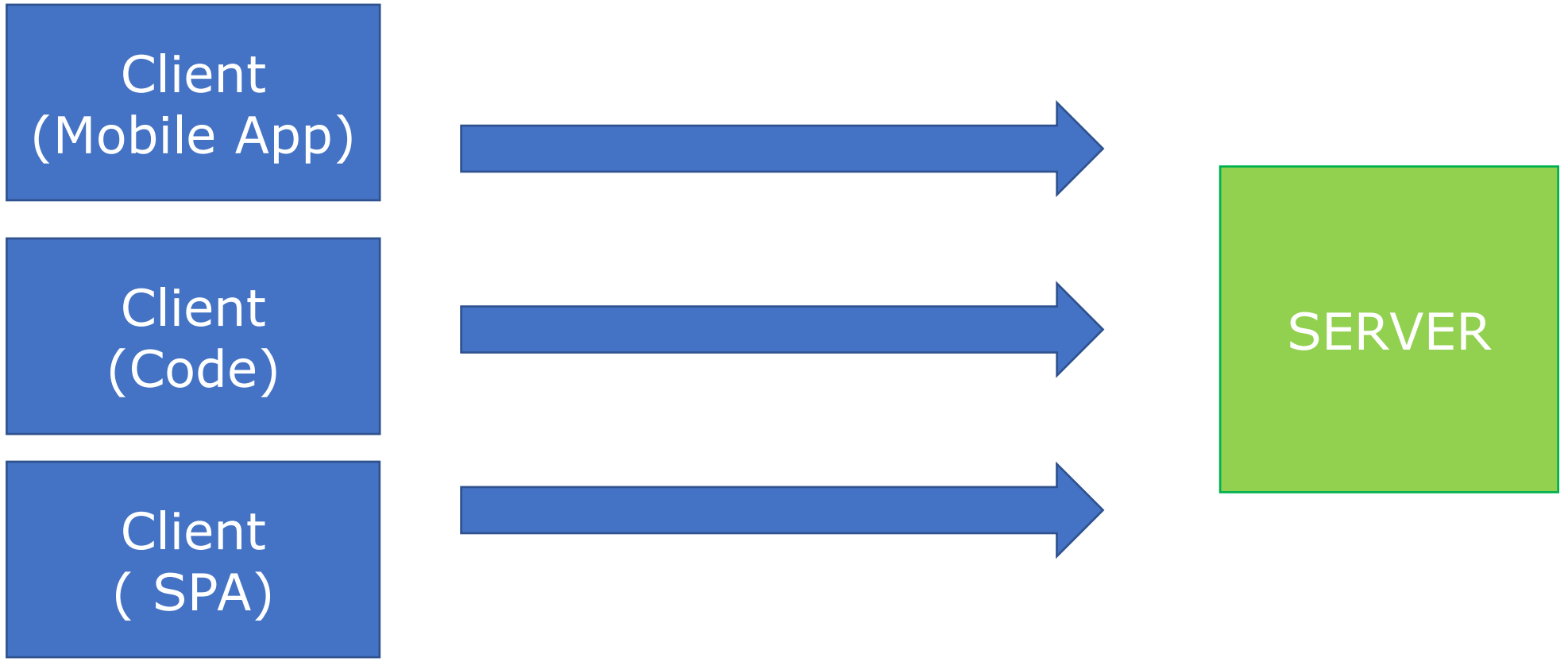
- API is an **A**pplication **P**rogramming **I**nterface
  - It is a set of rules that allow programs to talk to each other
  - Developers create API on the servers and allows the client to talk to it

- REST stands for **Re**presentational **S**tate **T**ransfer
  - A set of rules that developers follow when they create their API
    - One of these rules state you can get data via specific URL
      - Each URL becomes a **request**
      - The result becomes a **response**

Traditional HTML model



Mass server model



Resource Based URL

Action based URLs

- Focus on the action being performed
- Usually include a **verb**
- Often rely on **external sources** to identify the resource being acted on (e.g., session state)

- E,g, GET /register
  POST /register/edit?=123

Resource Based URLs

- Focus on the resource being acted on
- Usually consist of **nouns**
- Rely on **HTTP verbs** to define the action being performed (e.g., GET, PUT, POST, and DELETE)

- E.g. GET /item/123/form
  UPDATE /register/123

Representation

- Generally you deal with part of the resource **state**
  - Think about the template method of XSLTs when you grab a specific part of the XML document to process
  - This state is transfer between client and server

- Typically **states** are transferred as JSON or XML

Recommendations and constraints

- Uniform interface
  - Resources are identifies in requests
- Client-Server Architecture
  - Separation of concerns
- Stateless
  - No client-context is stored on the server
- Layered System
  - Intermediate servers can be used

Ilities from REST

- What we aim for in a long lasting web service:
- Scalability
- Simplicity
- Modifiability
- Visibility
- Portability
- Reliability

The Request

- Every request consist of four parts:
  - Endpoint
  - Method
  - Headers
  - Data/Body

Sample of RESTful API:

Github's API or look at the call on https://api.github.com

Twitter's API

Endpoint

- **Endpoint**(or route) is the URL you requested
  - **root-endpoint** is the starting point of the API
  - **path** determine the resource you are requesting
    - You can generally find available paths in the API documentation
    - In most API they will use (:) to represent variables

Endpoint

- **query parameters**
  - Not part of REST architecture but often used
  - Always being with (?) and separate with (&)

Written in the API:

```
GET /users/:username/repos
```

```
https://api.github.com/users/daddylonglegs/repos?sort=pushed
```

Method

- The **method** is the type of request you send to the server
  - GET
  - POST
  - PUT
  - PATCH
  - DELETE

- Methods provide a simple way of performing basic CRUD actions
  - Create, Read, Update and Delete

Calling CRUD Method

- GET (Read)
  - Request: Looks for the data
  - Respond: Sends the requested data back to you
- POST (Create)
  - Request: Creates a new entry in the database
  - Respond: The status of the creation
- PUT and PATCH (Update)
  - Request: Update an entry in the database
  - Respond: The status of the update

Calling CRUD Method

- DELETE (Delete)
  - Request: Delete an entry in the database
  - Respond: The status of the delete

- You can use most programming languages to speak to API

Written in the API:

```
GET /users/:username/repos
POST /user/repos

curl -X POST https://api.github.com/user/repos
```

Header

- **Headers** are used to provide information to both the client and server
- It can be used for many purposes:
  - authentication
  - providing information about the body content

- HTTP Headers are **property-value pairs** that are separated by a colon

```
curl -H "Content-Type: application/xml" https://api.github.com
```

Data/Body

- The **data** contains information you want to be sent to the server

```
curl –X POST <URL> -d name=value

(is the same)
curl --request POST <URL> -data name=value


curl –X POST https://request.in/applecider -d name=value
```

The application

```
curl -X POST https://api.mm.up.ac.za/tiny/profile \
       -H "Content-Type: application/json" \
       -d '{
              “Name":“Tiny",
               “Age":“109“
              }'
```

Known Challenges for REST

- Endpoint consensus
  - Declaration consistency
- API Versioning
  - Full system updates
- Authentication
  - Alternative methods of authorization
- Security
  - DDOS
- Multiple Requests and Unnecessary Data
  - Either request all or nothing

Theme 10.1 - End