Day 14

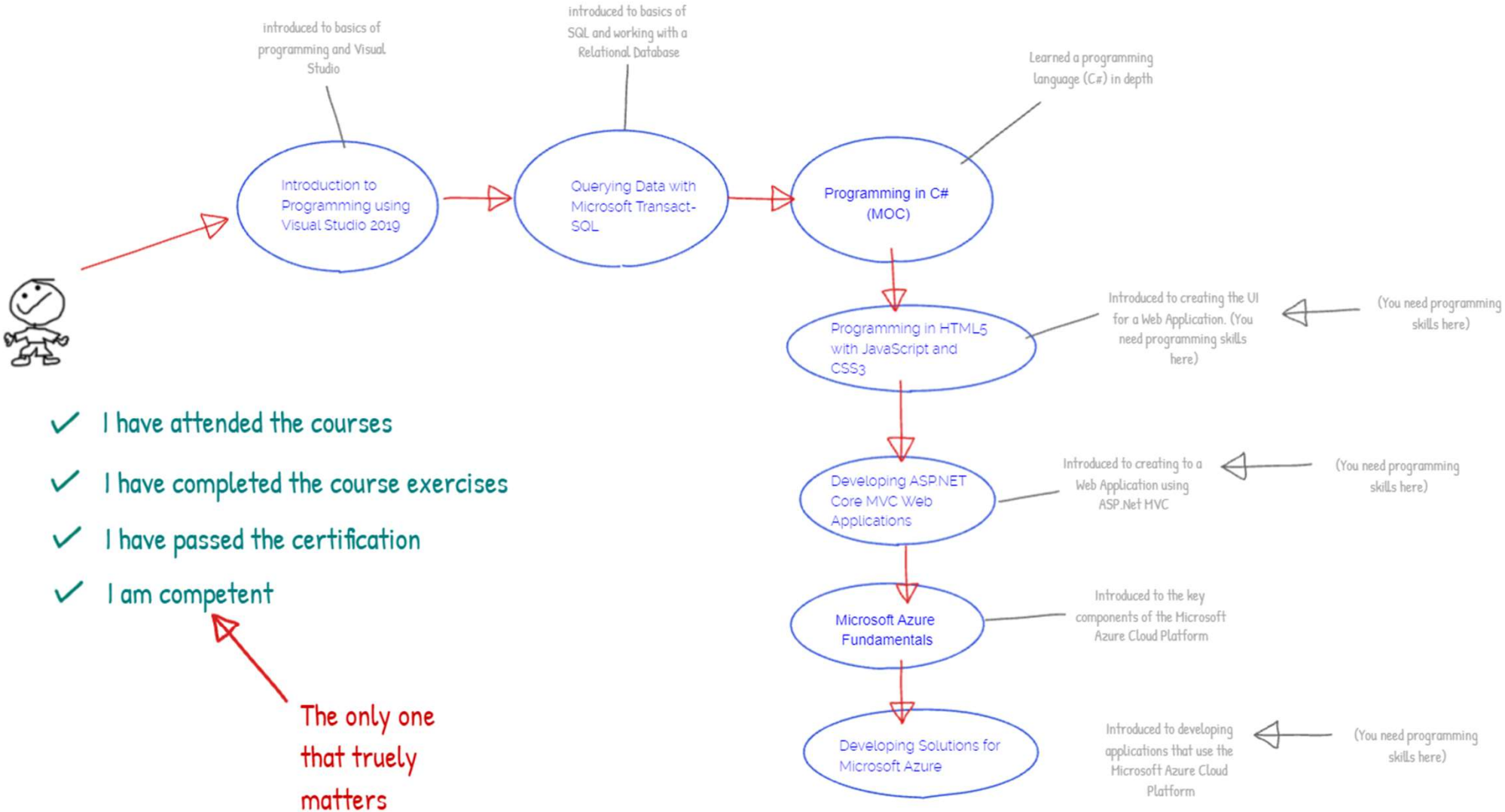
The one link you need to remember

https://ddls.to/20483

Ready?

Do this every day BEFORE the class starts (takes about 15 minutes) (http://ddls.to/everyday)

1. Launch Lab01.
2. Login to Lab01 as **Admin**.
3. While in the Lab01 environment,
   i. run **cmd.exe** from the Windows Start button.
   ii. Run the command **git clone --depth 1** https://github.com/Mark-AIICT/CAD-2.git **C:\Users\Admin\Desktop\MarksFiles**
   iii. Navigate to **C:\Users\Admin\Desktop\MarksFiles\setups**, then right-mouse click **bootstrap.cmd** and **run as administrator**
   iv. While it's running, Sign in to Visual Studio on the Lab Environment. You can use any Microsoft account.
   v. When the script end it reboots the Virtual Machine. That's necessary.
   vi. Save the lab. (the save link is at the top right of the screen in the dropdown menu)



- ✓ I have attended the courses
- ✓ I have completed the course exercises
- ✓ I have passed the certification
- ✓ I am competent

The only one that truely matters

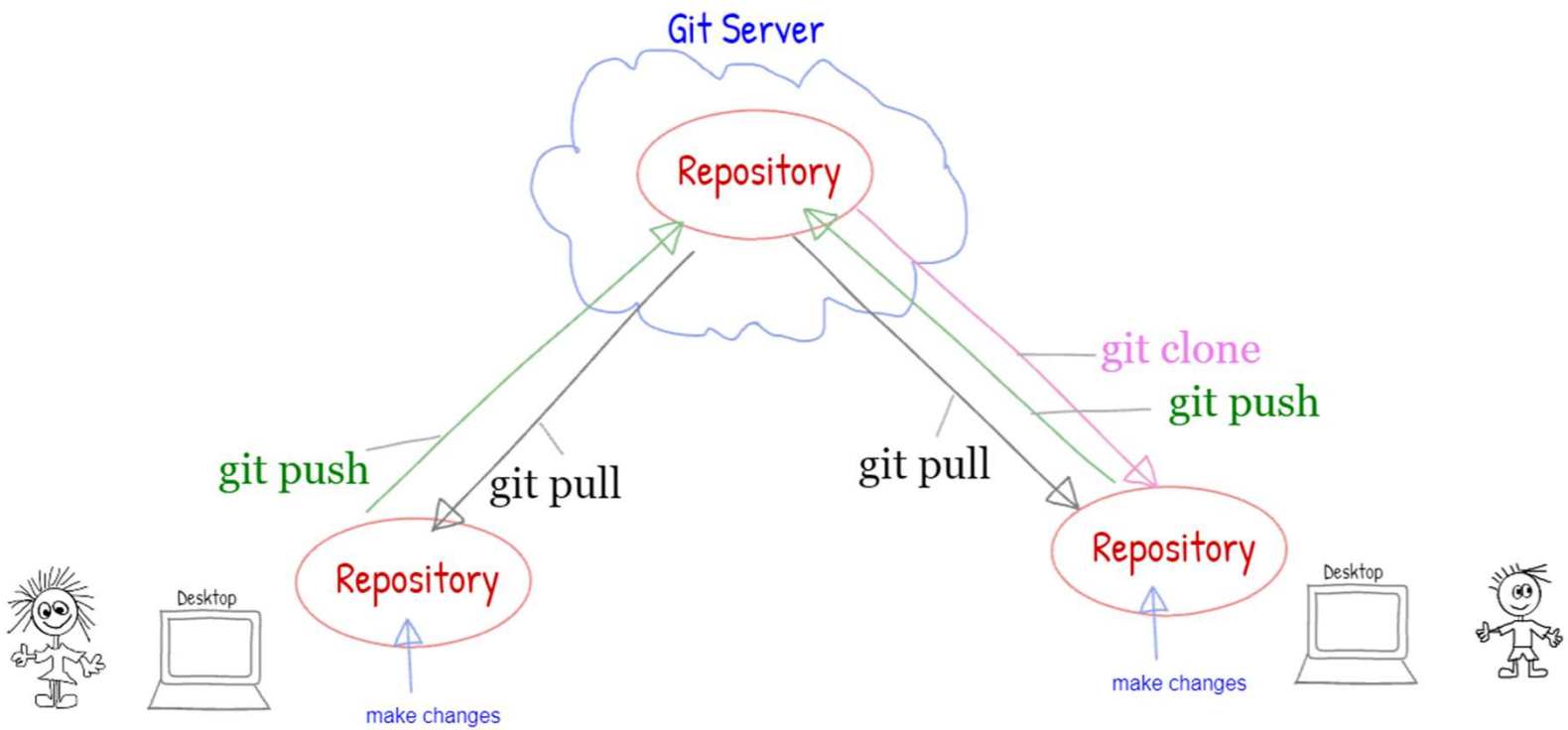
Course Outline

- Module 1: Review of Visual C# Syntax
- Module 2: Creating Methods, Handling Exceptions, and Monitoring Applications
- Module 3: Basic Types and Constructs of Visual C#
- Module 4: Creating Classes and Implementing Type-Safe Collections
- Module 5: Creating a Class Hierarchy by Using Inheritance
- Module 6: Reading and Writing Local Data
- Module 7: Accessing a Database
- Module 8: Accessing Remote Data (I'm replacing this with a better module)
- Module 9: Designing the User Interface for a Graphical Application
- Module 10: Improving Application Performance and Responsiveness
- GIT
- REST + WPF + Sharpen C# Skills
- Config Files
- .Net Framework Vs .Net Core Vs .net standard Vs .Net
- VSCode Vs Visual Studio
- Nuget
- Module 11: Integrating with Unmanaged Code?
- Module 12: Creating Reusable Types and Assemblies?
- Module 13: Encrypting and Decrypting Data?

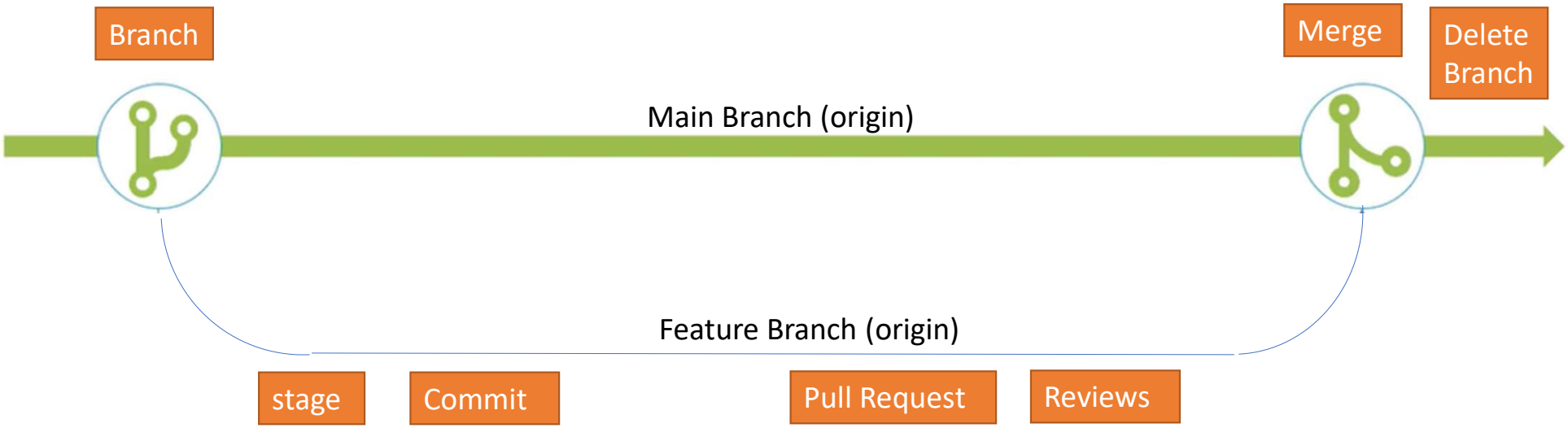
Key Recollections from earlier

Class
Inheritance
Method
Property
Field
Public
Private
Protected
Internal
Interface
Delegate
Event
Lambda
Task

Cloning repo's and collaborating



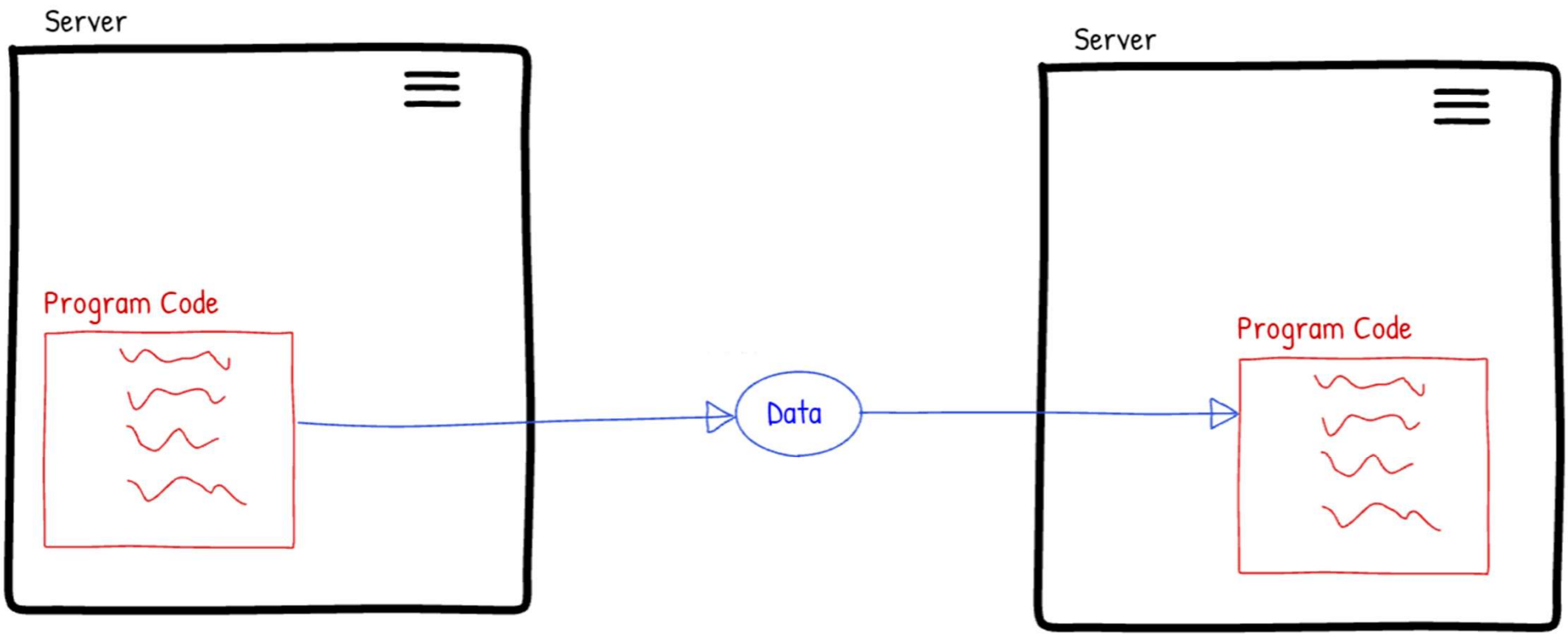
Our Branching strategy



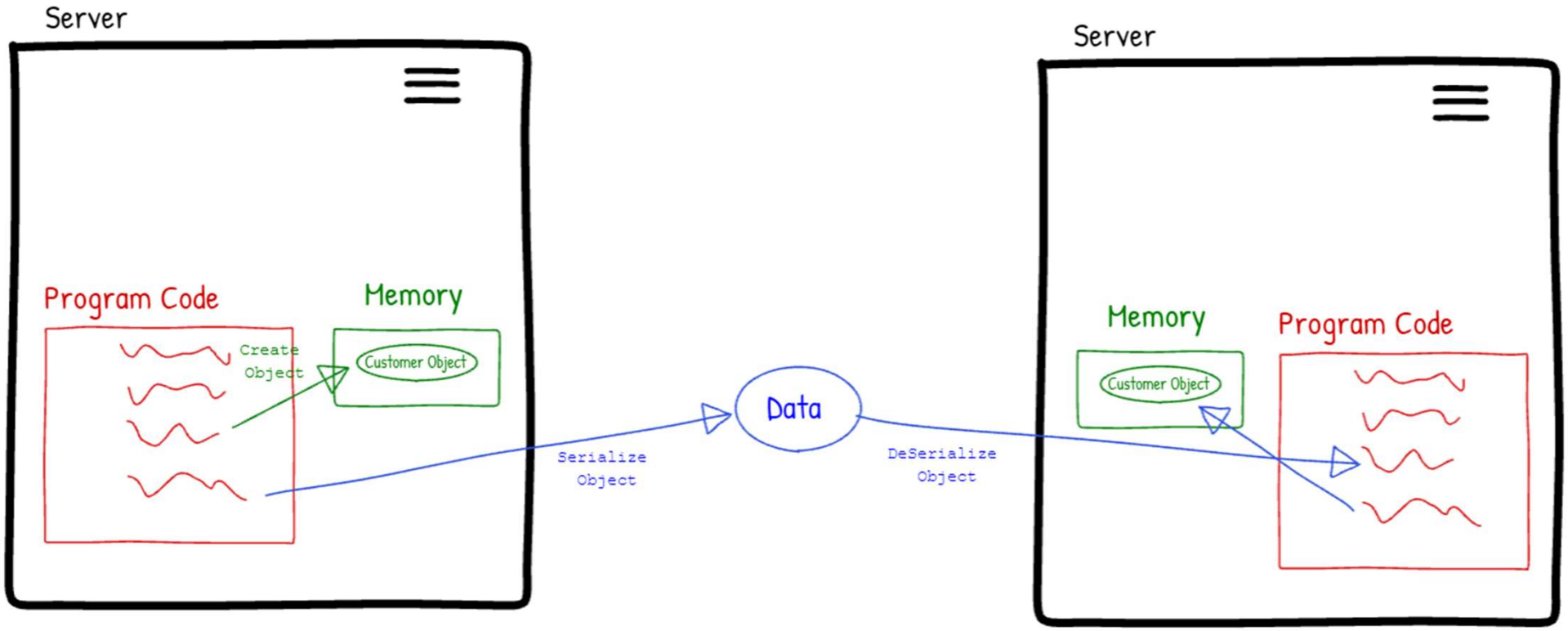
REST

- Representational State Transfer

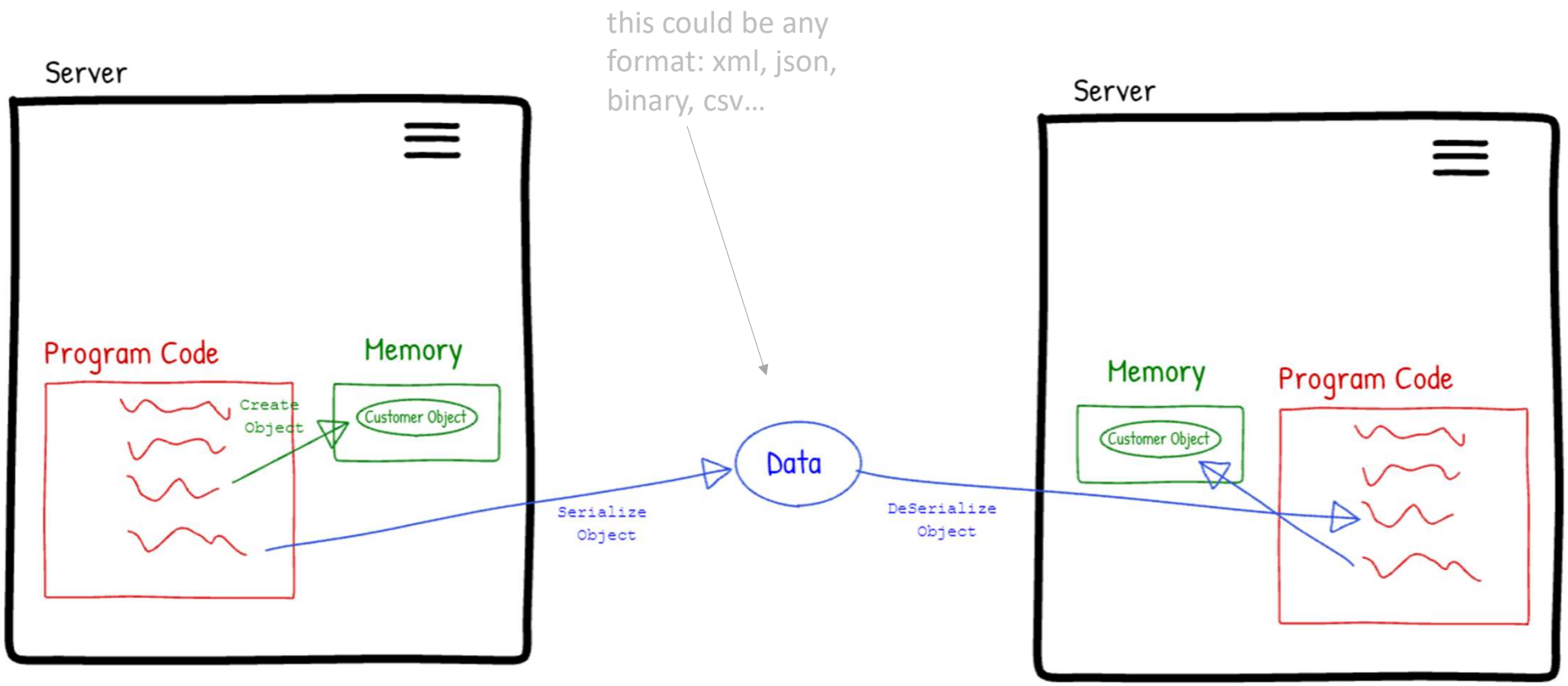
How does a program send an object to another program?



How does a program send an object to another program?



How does a program send an object to another program?



What is JSON?

JSON = JavaScript Object Notation

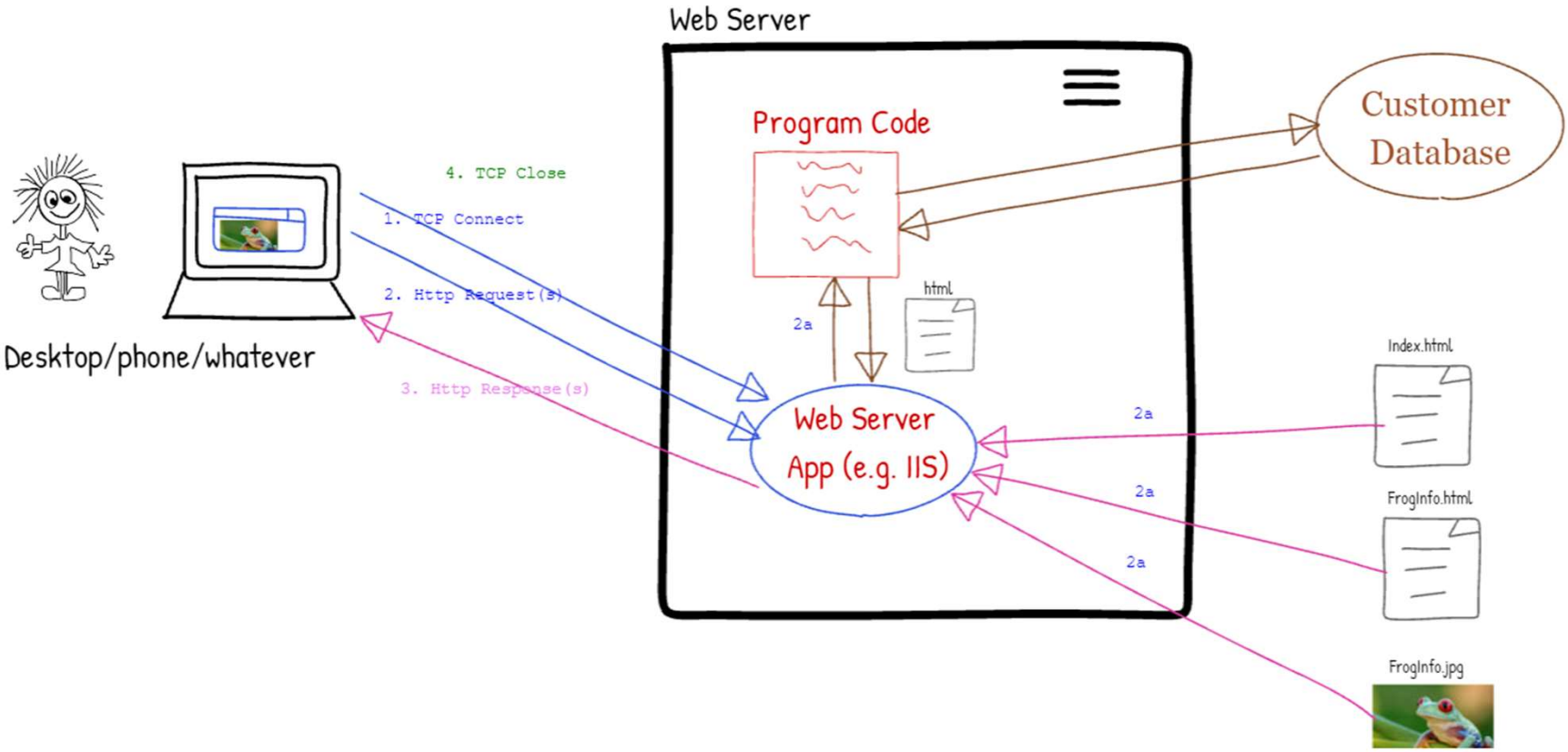
JSON is typically used to hold structured data

JSON has a few key concepts:

- Key-Value pairs
- DataTypes
  - strings
  - numbers
  - objects
  - arrays
  - Booleans (true or false)
  - null
- Nested Objects
- Nested Arrays

```
{
 "first_name" : “Hunstman",
 "last_name" : “Spider",
 "location" : “In Your Car",
 "websites" : [
  {
   "description" : "work",
   "URL" : "https://aiict.edu.au/"
  },
  {
   "desciption" : "tutorials",
   "URL" : "https://www.aiict.edu.au/community/tutorials"
  }
 ],
 "social_media" : [
  {
   "description" : "twitter",
   "link" : "https://twitter.com/aiict"
  },
  {
   "description" : "facebook",
   "link" : "https://www.facebook.com/aiictCloudHosting"
  },
  {
   "description" : "github",
   "link" : "https://github.com/aiict"
  }
 ]
}
```

Key elements of the 'the Web'



What is an API?

API = Application Programmer Interface

Not just seen in Web Applications

Is for *Application Programmers* rather that *Users*

What is REST?

REST is not a protocol

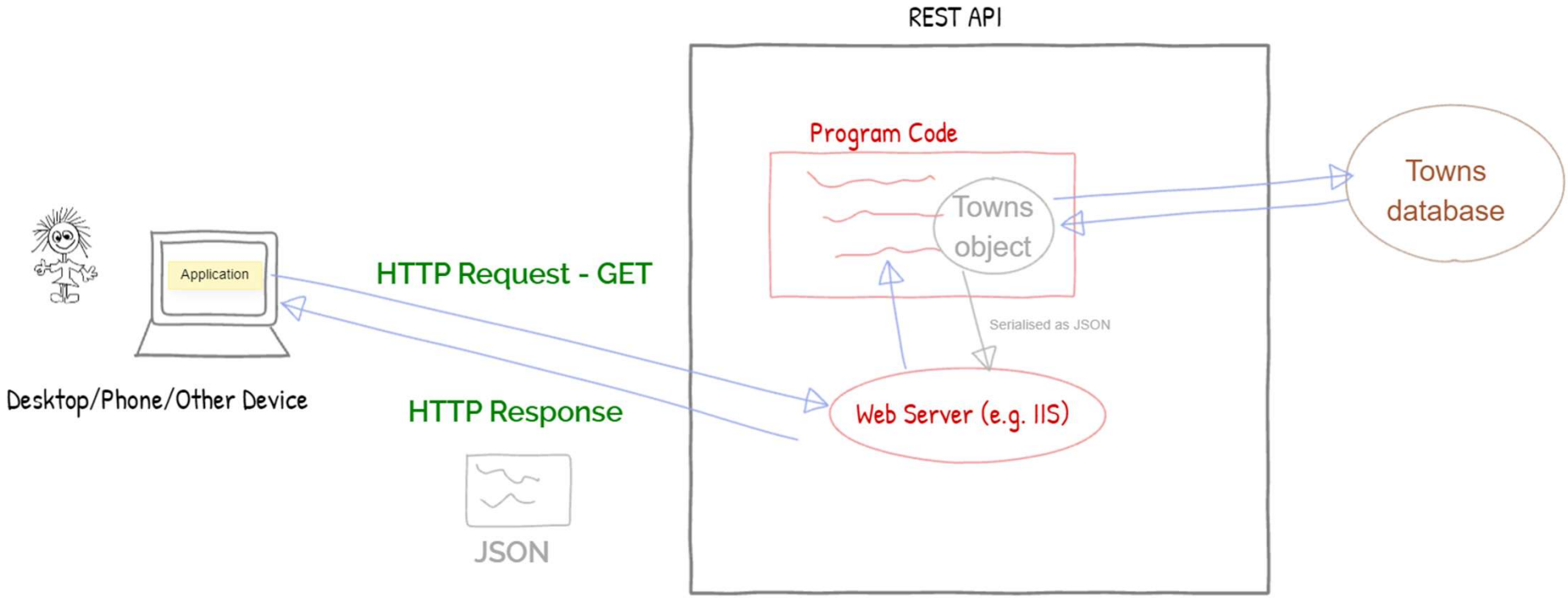
REST is not a standard

REST is not a new idea

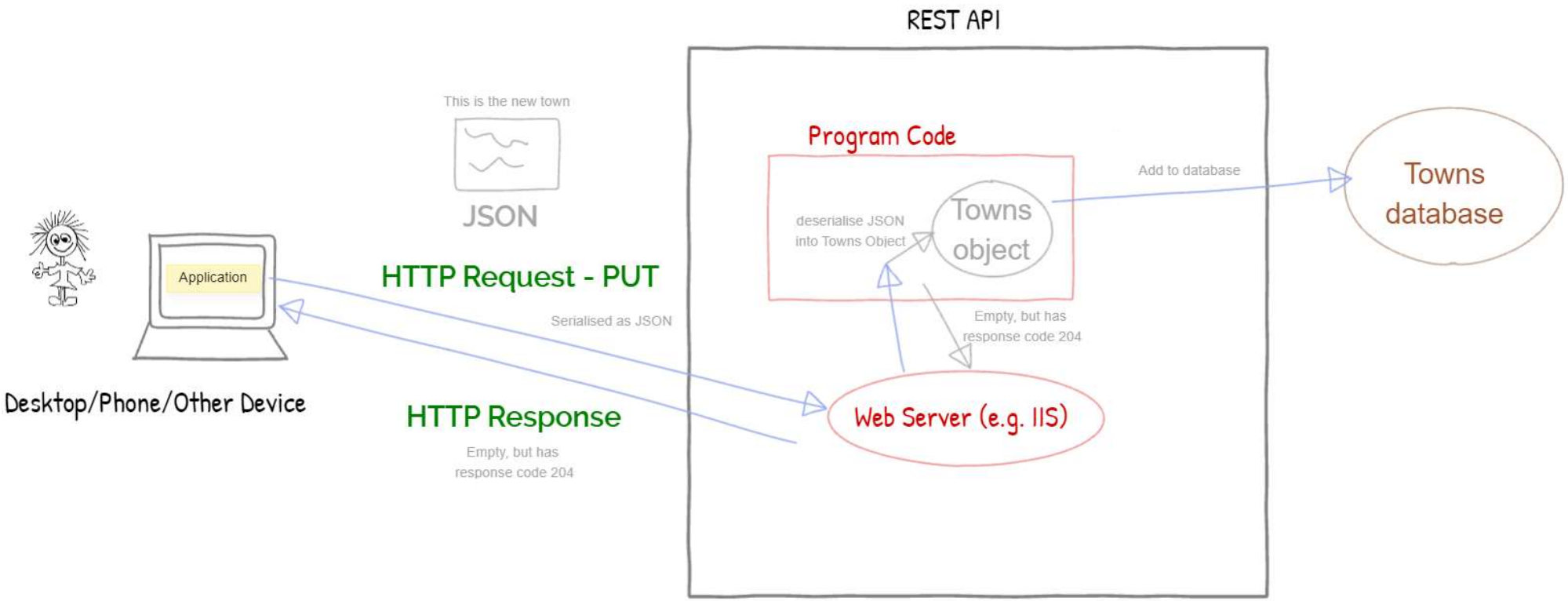
REST is not a formula

REST is a set of semantics, REST is a style

What is REST?


REST API
Program Code
Towns object
Towns database
Serialised as JSON
Application
HTTP Request - GET
Desktop/Phone/Other Device
HTTP Response
Web Server (e.g. IIS)
JSON

REST PUT Example

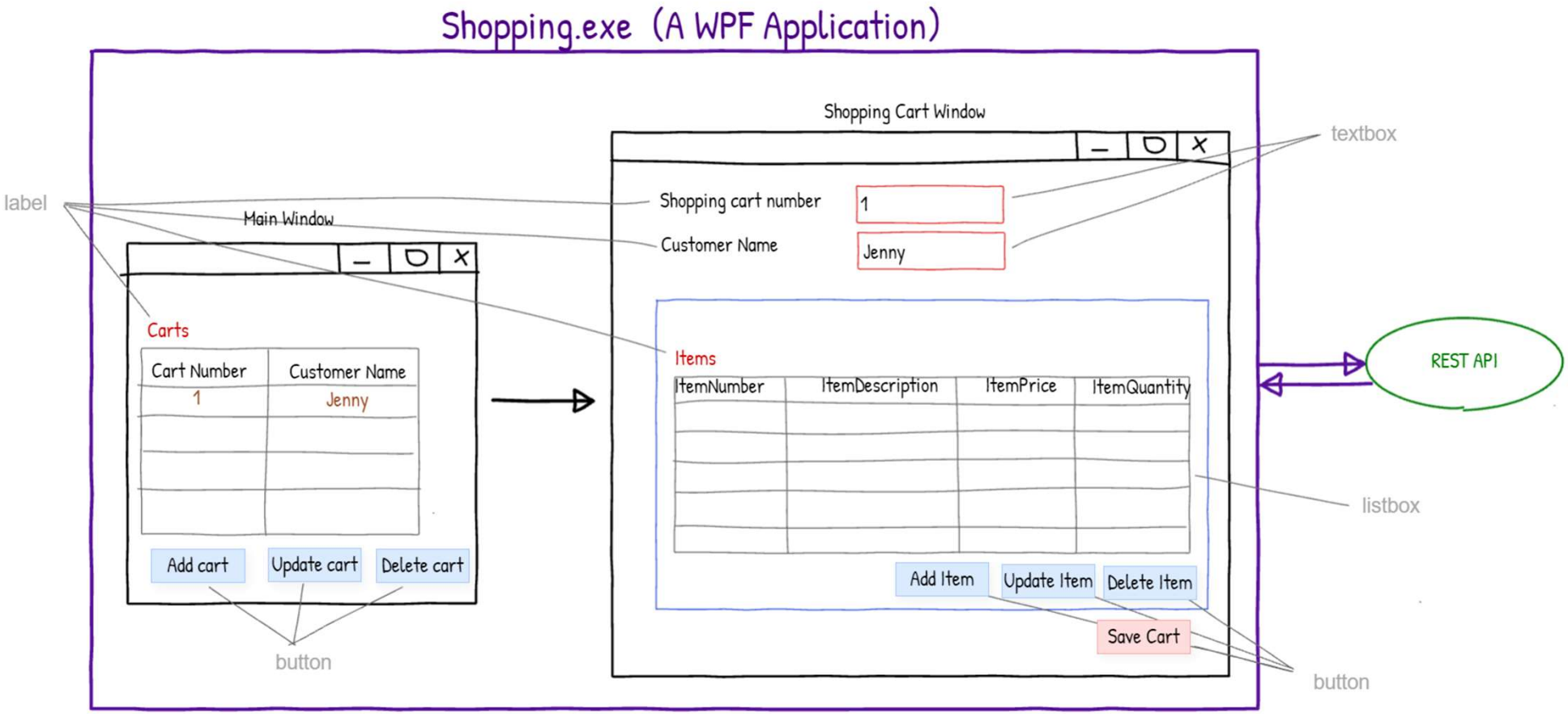


How do I call a REST API?

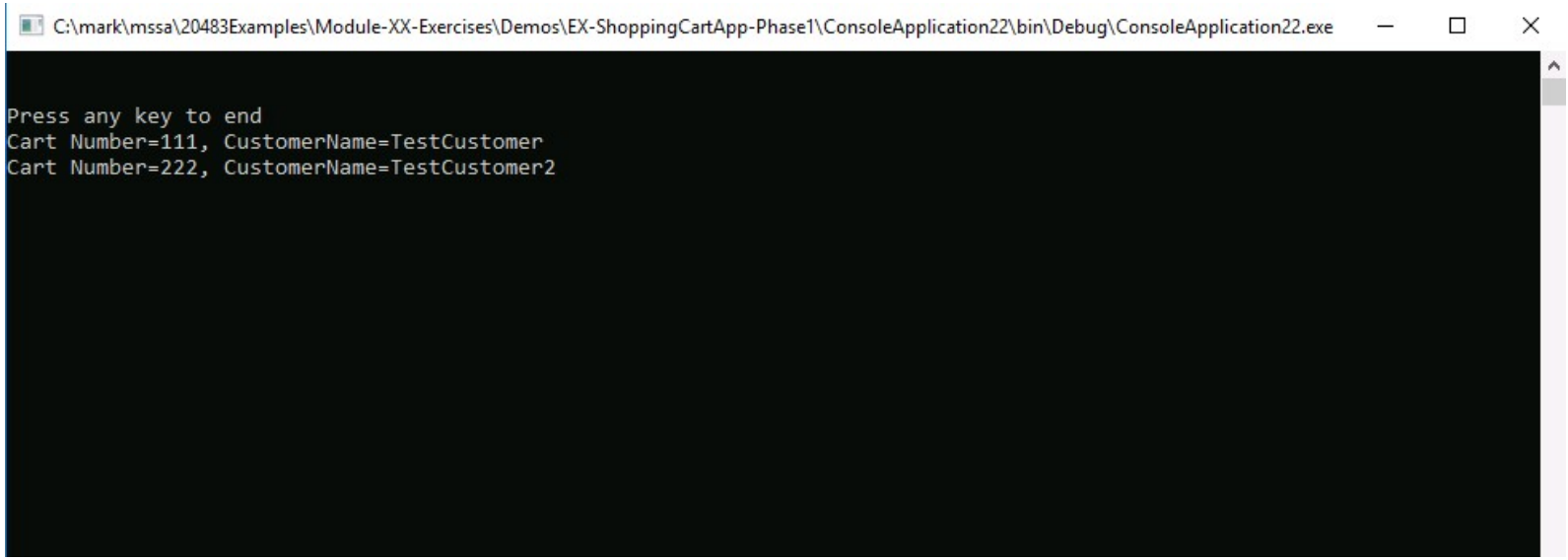
The newer way

```
static void Main(string[] args)
        {
                Action task = new Action(RestCalls);
                Task.Run(task).Wait();
        }

static async void RestCalls()
        {
                HttpClient client = new HttpClient();

                Console.WriteLine("Retrieving....\n\n");
                var response = await client.GetAsync("uri goes here");
                var s = await response.Content.ReadAsStringAsync();

                Console.WriteLine($"s={s}");
        }
}
```

Exercise


Shopping.exe (A WPF Application)
Main Window
Carts
Cart Number
Customer Name
1
Jenny
Add cart
Update cart
Delete cart
button
label
Shopping Cart Window
Shopping cart number
1
Customer Name
Jenny
textbox
Items
ItemNumber
ItemDescription
ItemPrice
ItemQuantity
listbox
Add Item
Update Item
Delete Item
Save Cart
button
REST API

Exercise Phase - 1



Exercise Phase – n – Our Current Branching Strategy

Create new local Branch for new work

1. git branch *mynewwork*
2. git checkout *mynewwork*
3. Do work
4. Stage (git add), Commit (git commit)
5. git push

Clean-up your Remote Branch

1. Merge into the main branch on GitHub with a PR

Clean-up your local branches

1. git checkout main
2. git remote prune origin
3. git branch
4. *for each local branch where you have finished work*
   git branch –delete *branchname*

Exercise Phase – n – Our new Branching Strategy

Easy As bro!

1. git checkout main
2. Do work
3. Commit
4. Stage (git add), Commit (git commit)
5. git push

Take care not to change other people's folders

- To do on your lab01 VM
  - Set git global config

```
git config --global user.name “fred”
git config --global user.email fred@work.com
```

  - Clone the exercises repo