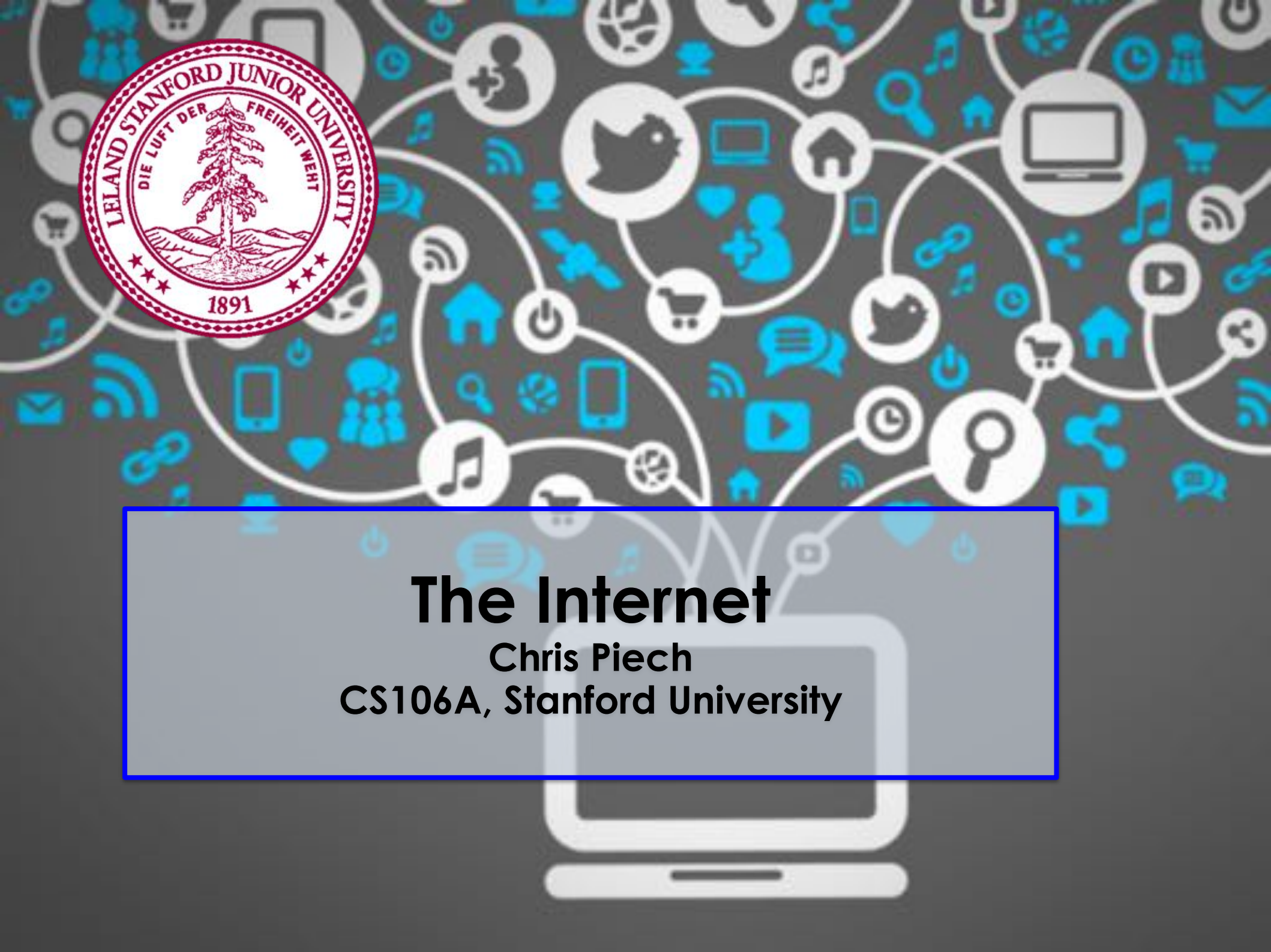


The Internet

Chris Piech
CS106A, Stanford University

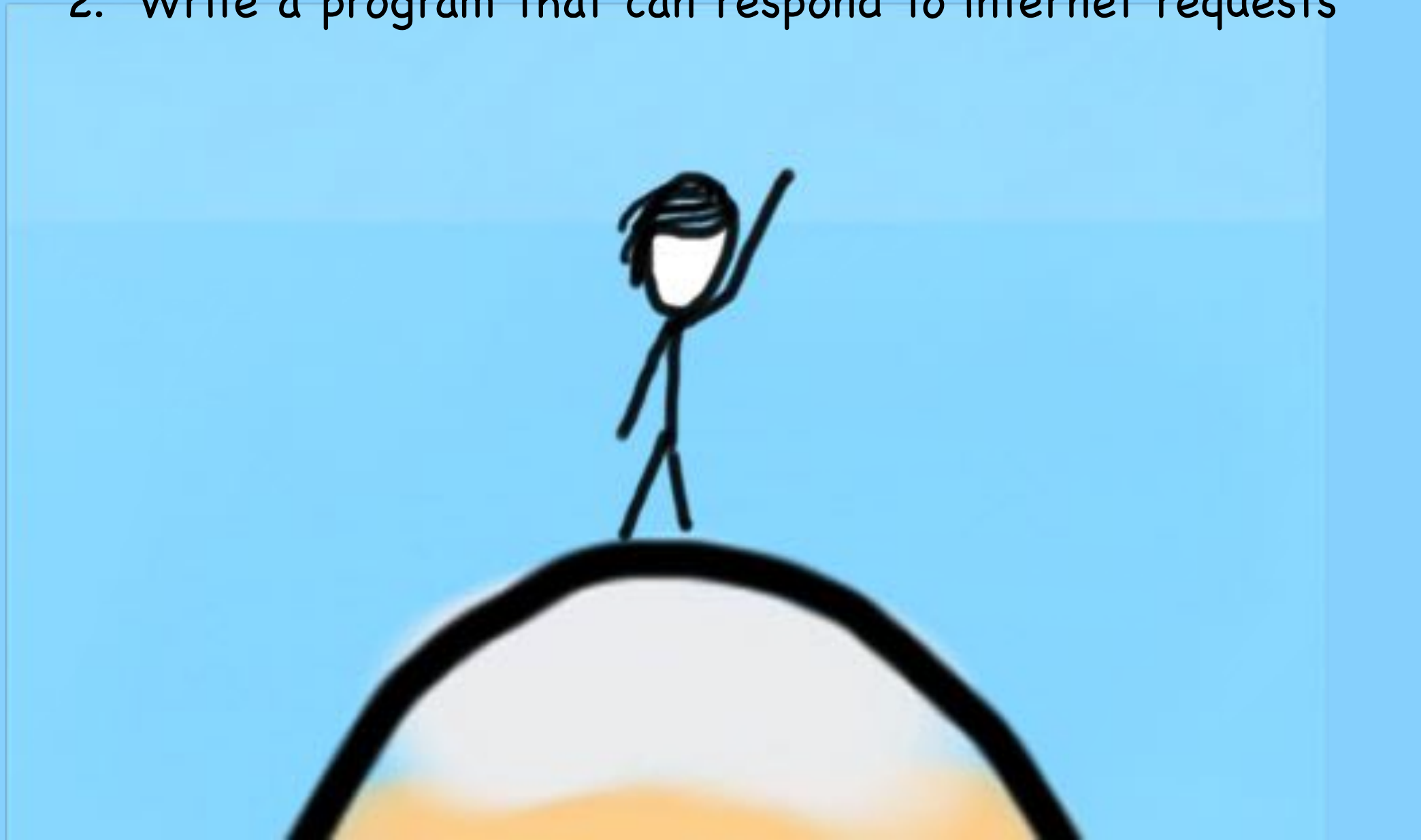
I came here to make friends
and program the internet...

... And I already made friends.

For the second time ever in CS106A:

Learning Goals

1. Write a program that can make internet requests
2. Write a program that can respond to internet requests



How does your phone communicate with facebook?

The Java program on your **phone** talks to the Java program at **Facebook**

Face Book Server

* Android phones run Java. So do facebook servers

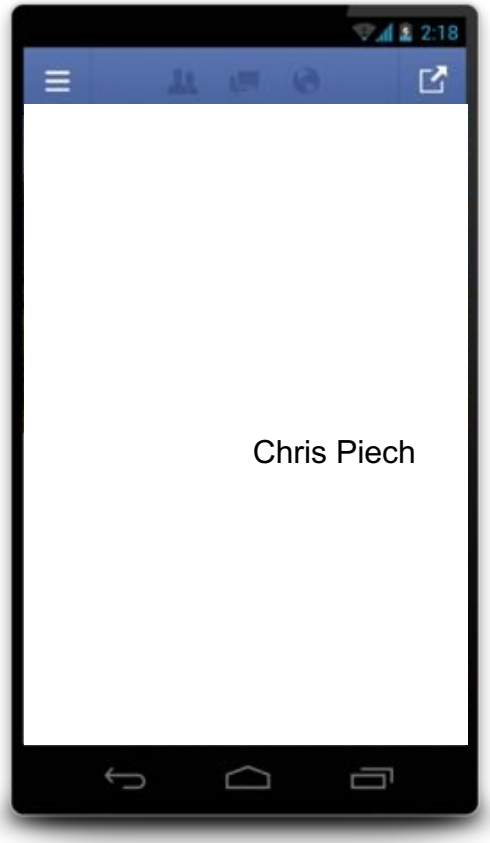
Face Book Server

Is this legit?

piech@cs.stanford.edu
is now logged in

Face Book Server

Send me the full name for
piech@cs.stanford.edu



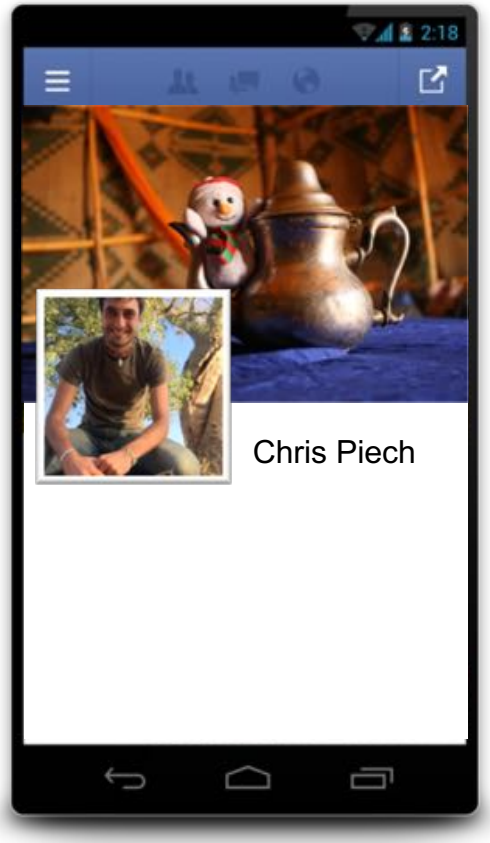
"Chris Piech"

Face Book Server

Send me the cover photo for piech@cs.stanford.edu

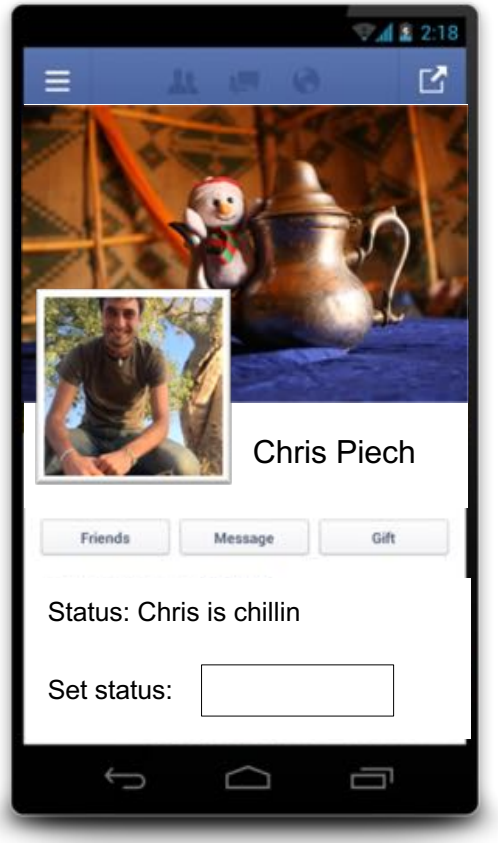
Face Book Server

Send the profile photo for piech@cs.stanford.edu



Face Book Server

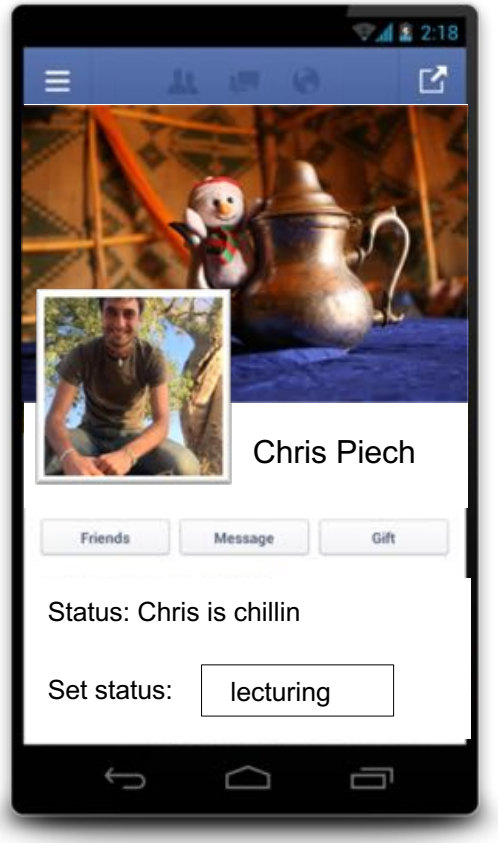
Send the status for piech@cs.stanford.edu



"chillin"

Face Book Server

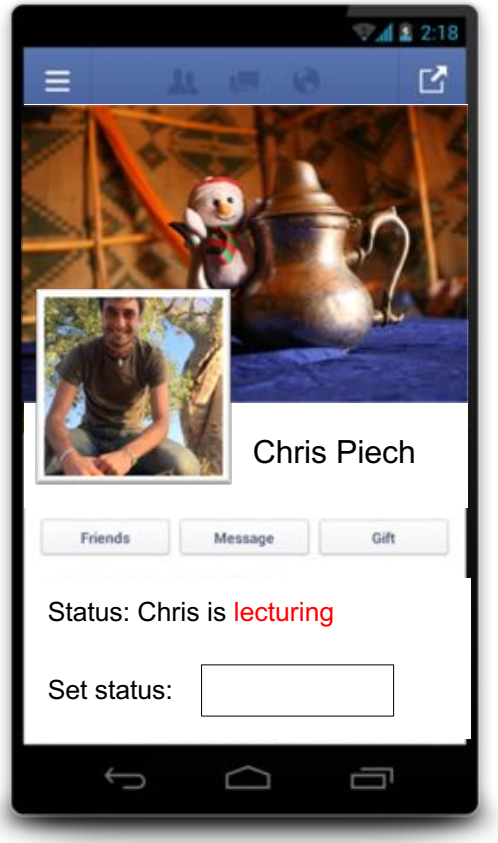
Set the status for piech@cs.stanford.edu to be "lecturing"



"success"

Face Book Server

Send me the status for piech@cs.stanford.edu



"lecturing"

Background: The Internet

The internet is just many programs sending messages (as *Strings*)

Thanks Nick for the teaching YEAH

Background: The Internet

The internet is just many programs sending messages (as *Strings*)

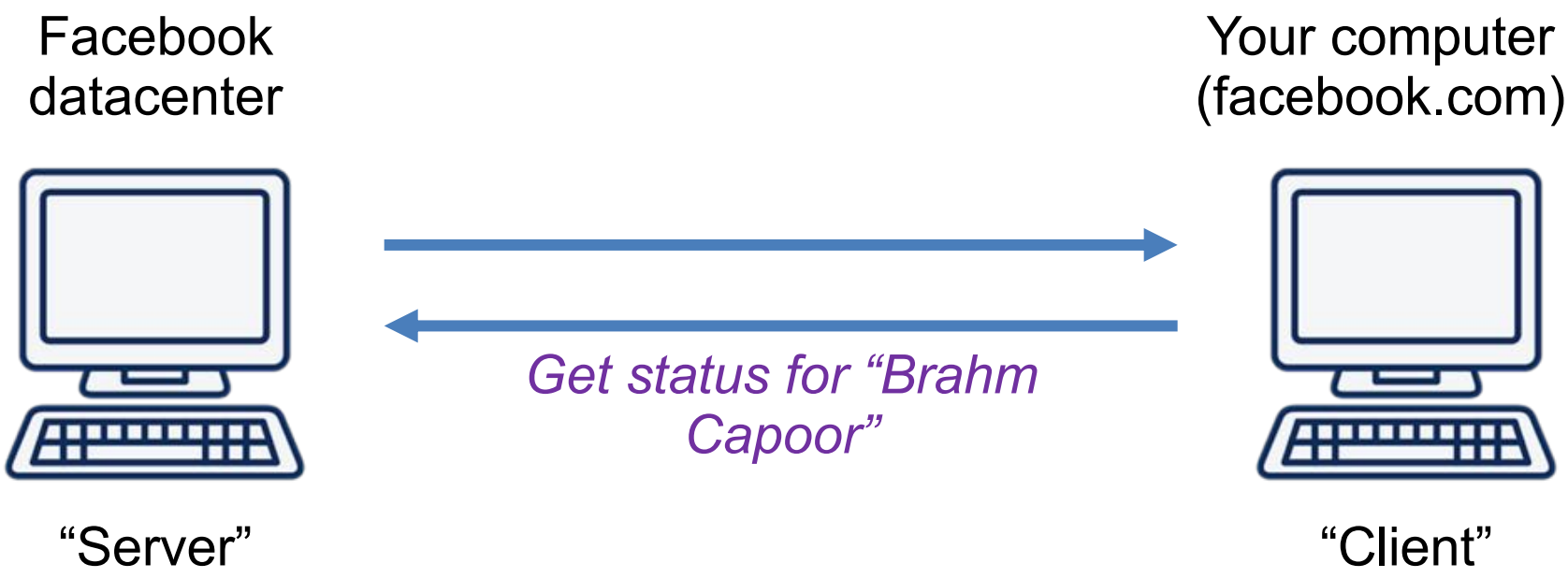
Thanks Nick for the teaching YEAH

Background: The Internet

The internet is just many programs sending messages (as *Strings*)

Thanks Nick for the teaching YEAH

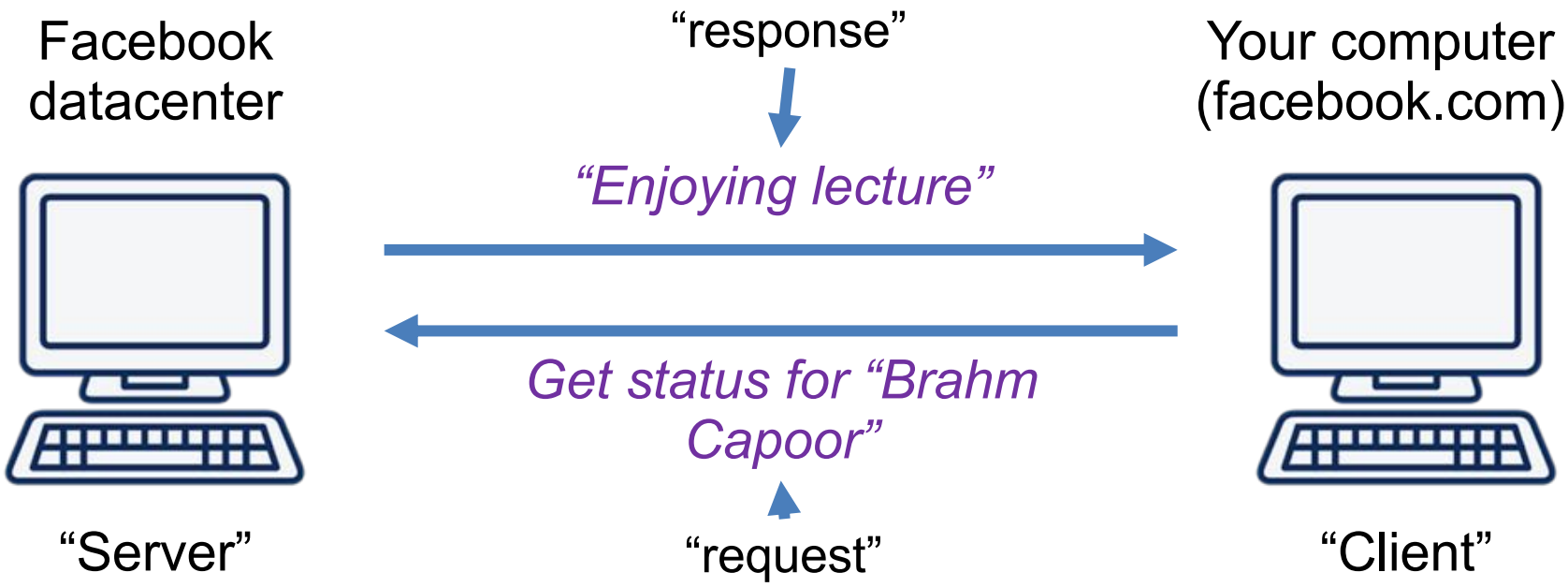
Background: The Internet



The internet is just many programs sending messages (as *Strings*)

Thanks Nick for the teaching YEAH

Background: The Internet



The internet is just many programs sending messages (as *Strings*)

Thanks Nick for the teaching YEAH

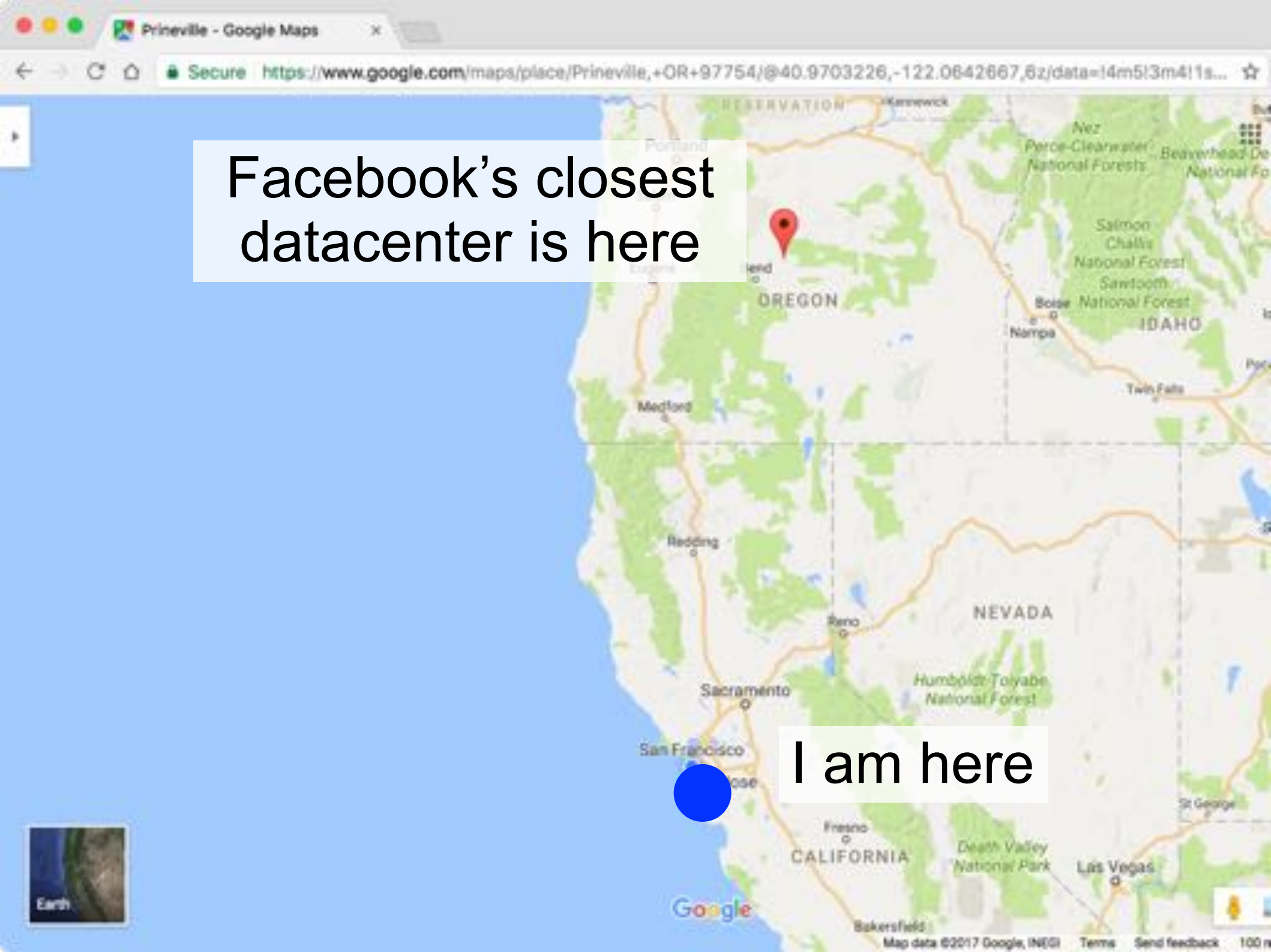
There are two types of internet programs. Servers and Clients

Internet 101

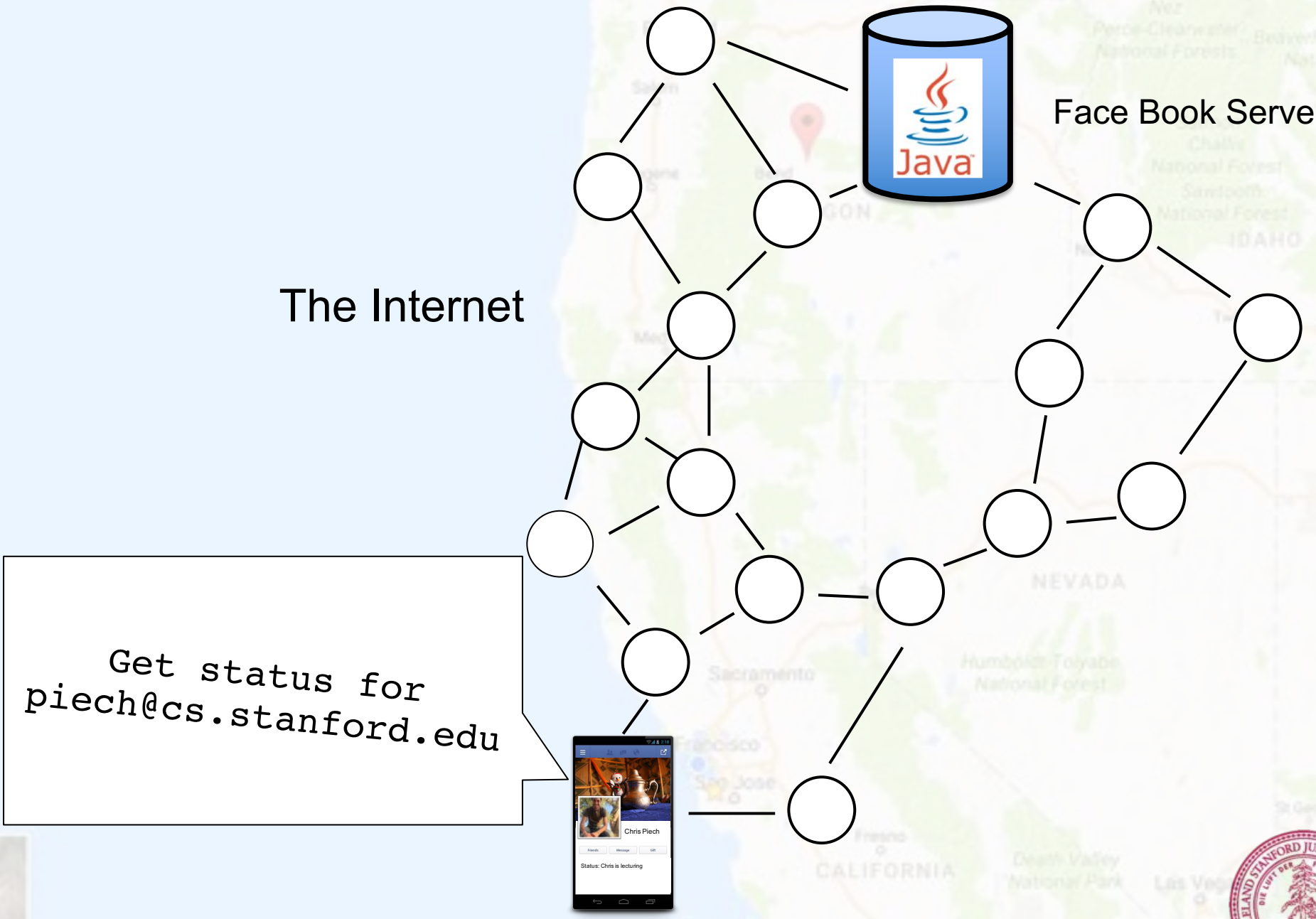
Servers are computers (running code)

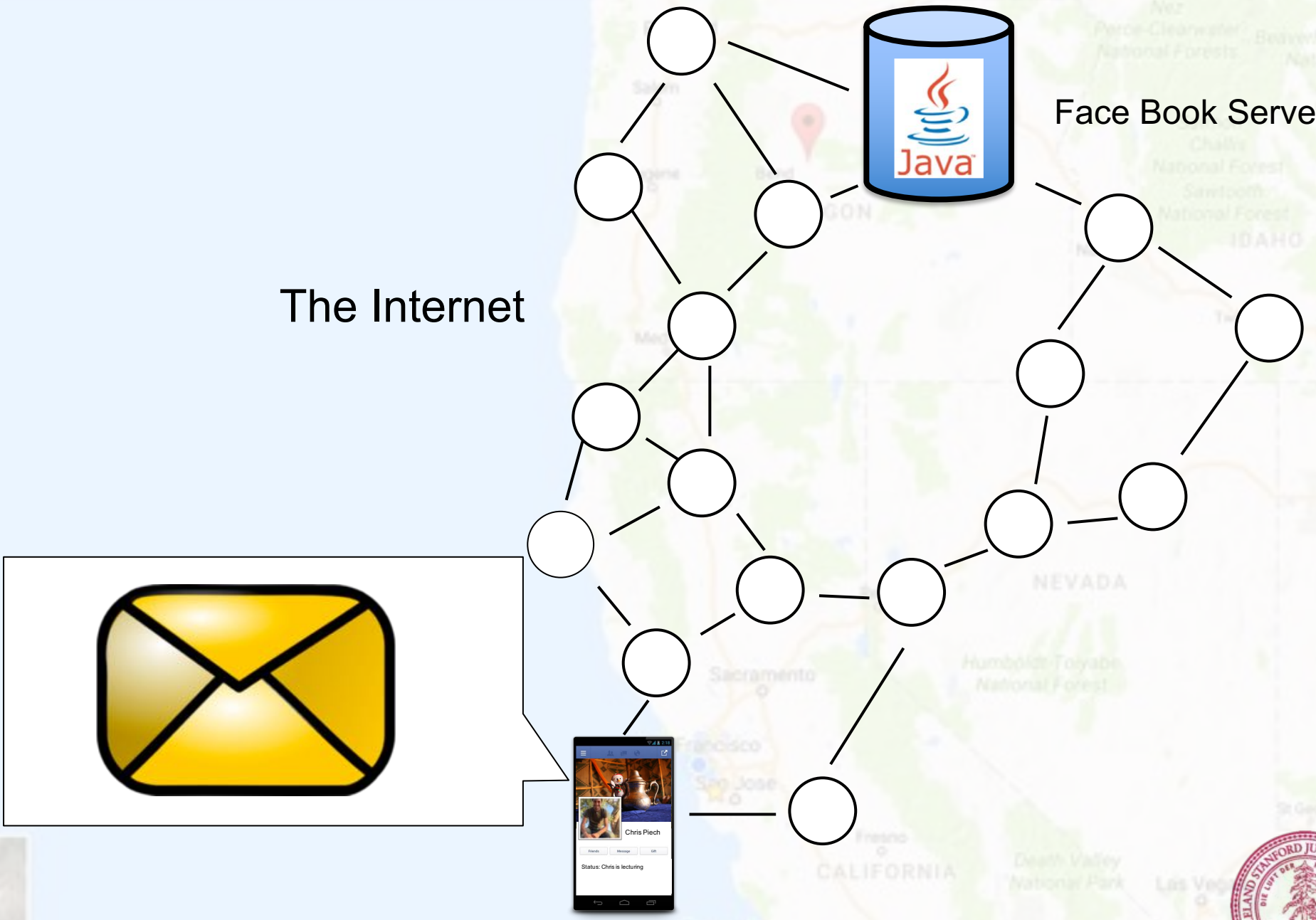
Face Book Server

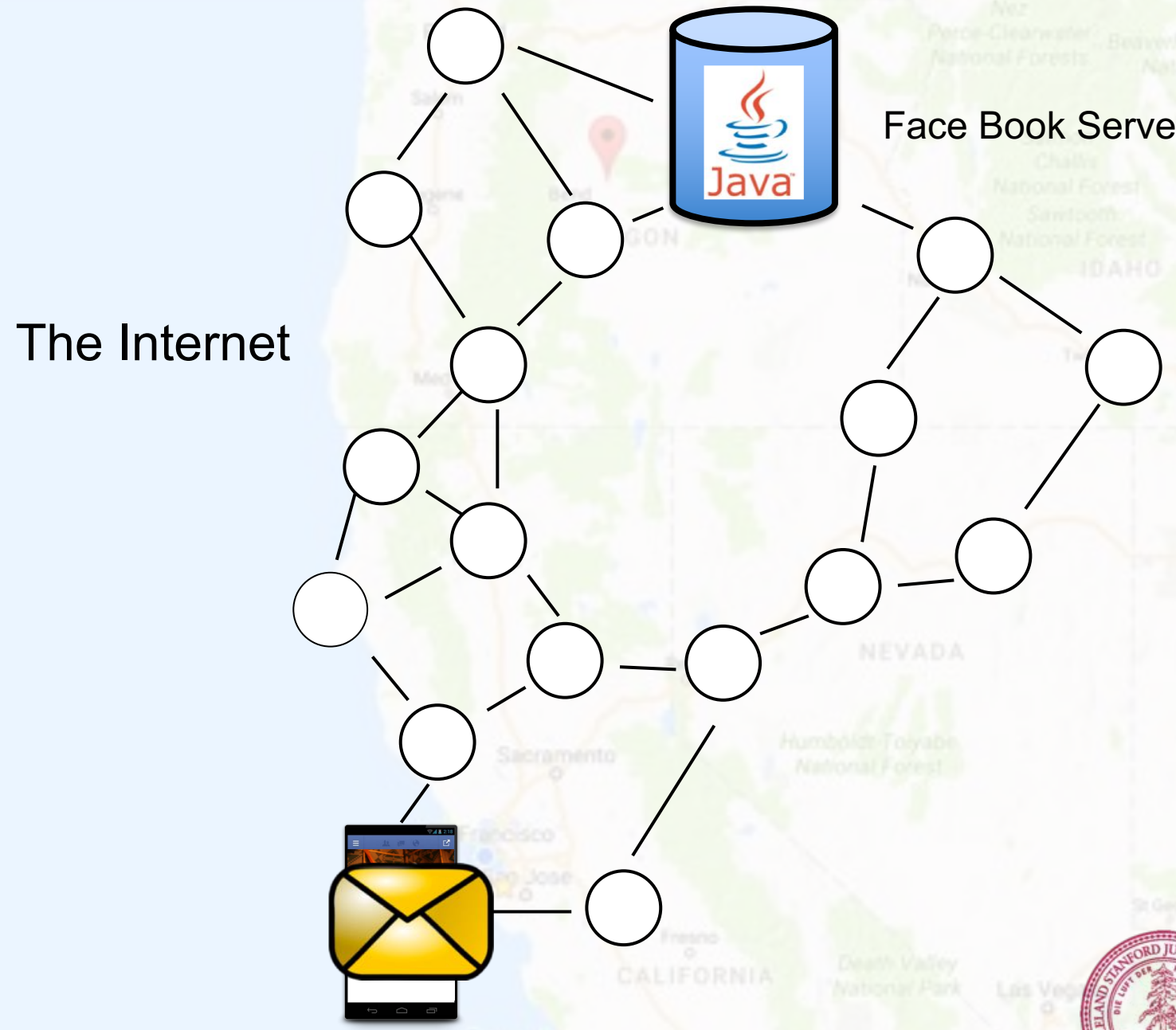
=

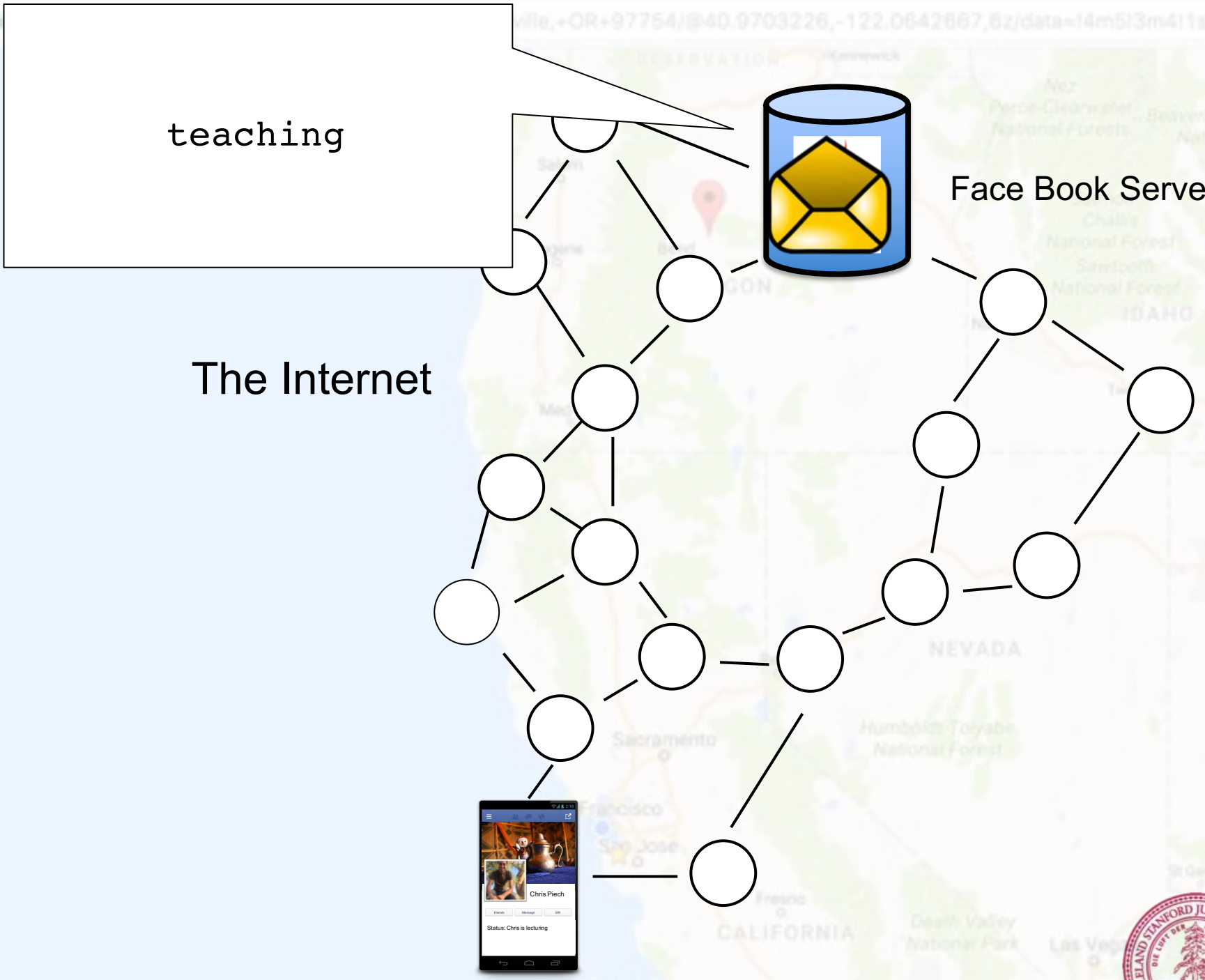

Prineville - Google Maps
Secure https://www.google.com/maps/place/Prineville,+OR+97754/@40.9703226,-122.0642667,6z/data=!4m5!3m4!1s...
Facebook's closest datacenter is here
I am here
OREGON
IDAHO
NEVADA
CALIFORNIA
Portland
Medford
Redding
Reno
Sacramento
San Francisco
Fresno
Las Vegas
Boise
Nampa
Twin Falls
St George
Bakersfield
Nez Perce-Clearwater National Forests
Salmon Challis National Forest
Sawtooth National Forest
Humboldt-Toiyabe National Forest
Death Valley National Park
Earth
Google
Map data ©2017 Google, INEGI
Terms
Send feedback

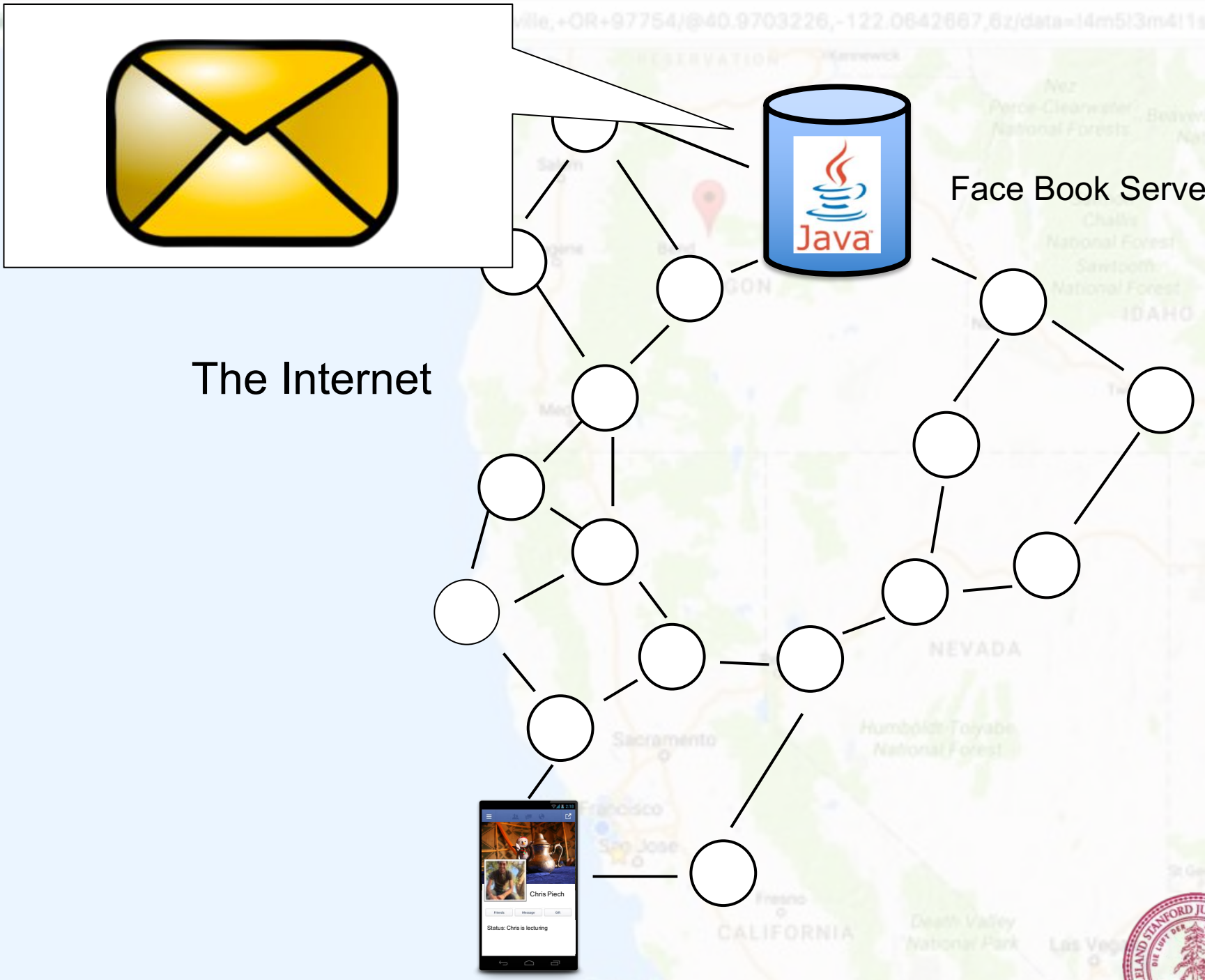

Face Book Server
The Internet
Java
Chris Piech
Status: Chris is lecturing


Face Book Server
The Internet
Get status for
piech@cs.stanford.edu
Chris Piech
Status: Chris is lecturing

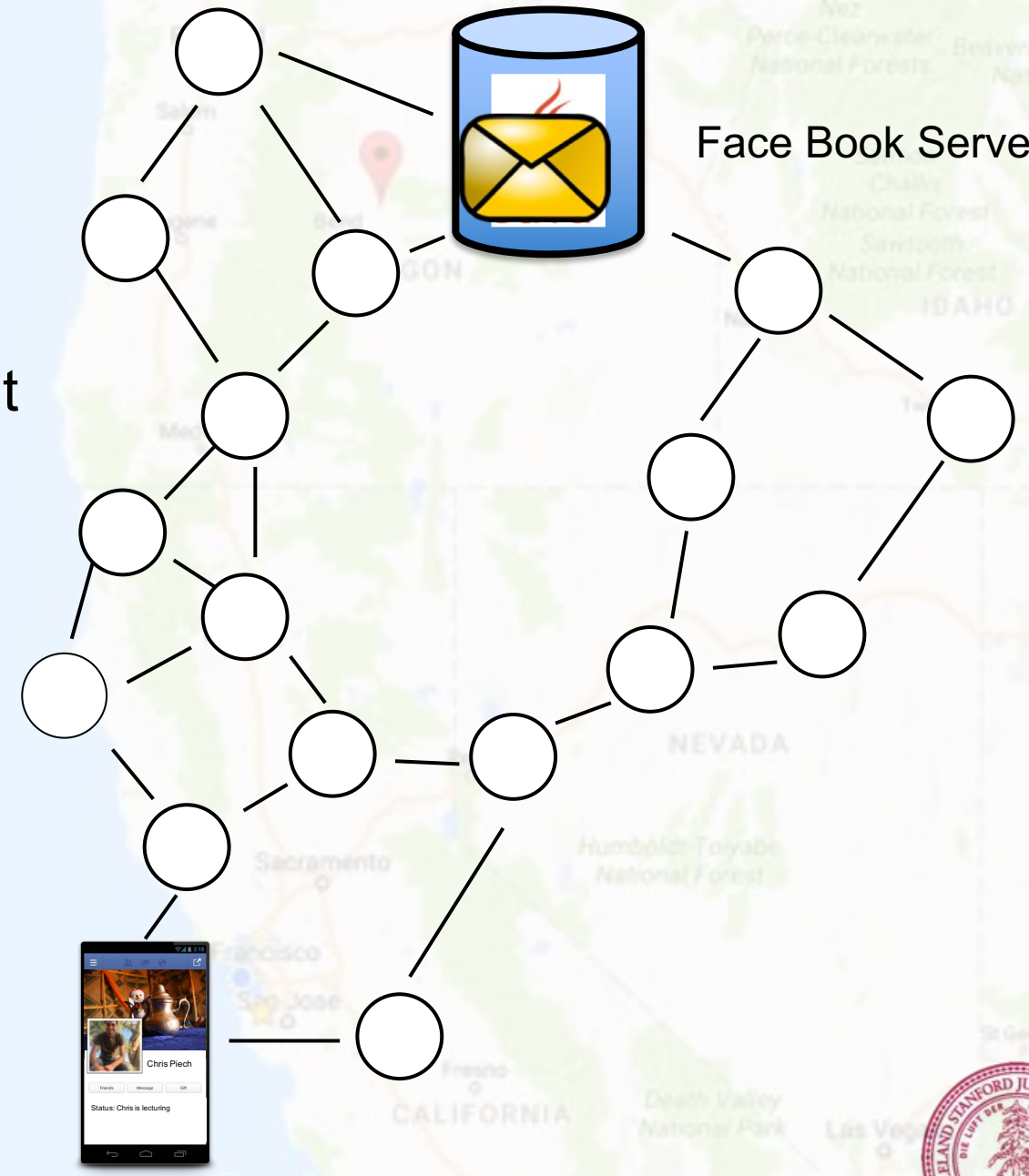

Face Book Server
The Internet
Chris Piech
Status: Chris is lecturing


Face Book Server
The Internet


teaching
Face Book Server
The Internet
Chris Piech
Status: Chris is lecturing


Face Book Server
The Internet
Chris Piech
Status: Chris is lecturing

`teaching`


Face Book Server
The Internet
Chris Piech
Status: Chris is lecturing

Many computers can connect
to the same server

The Internet

Facebook datacenter

"Server"

REQUEST

RESPONSE

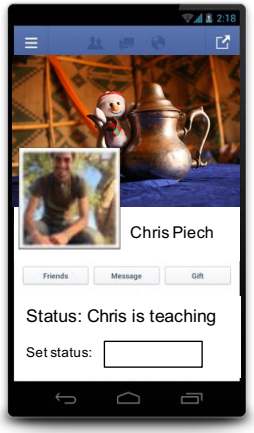
RESPONSE

REQUEST

Tate Ole Keko's computer (facebook.com)

"Client"

Chris' phone (facebook app)

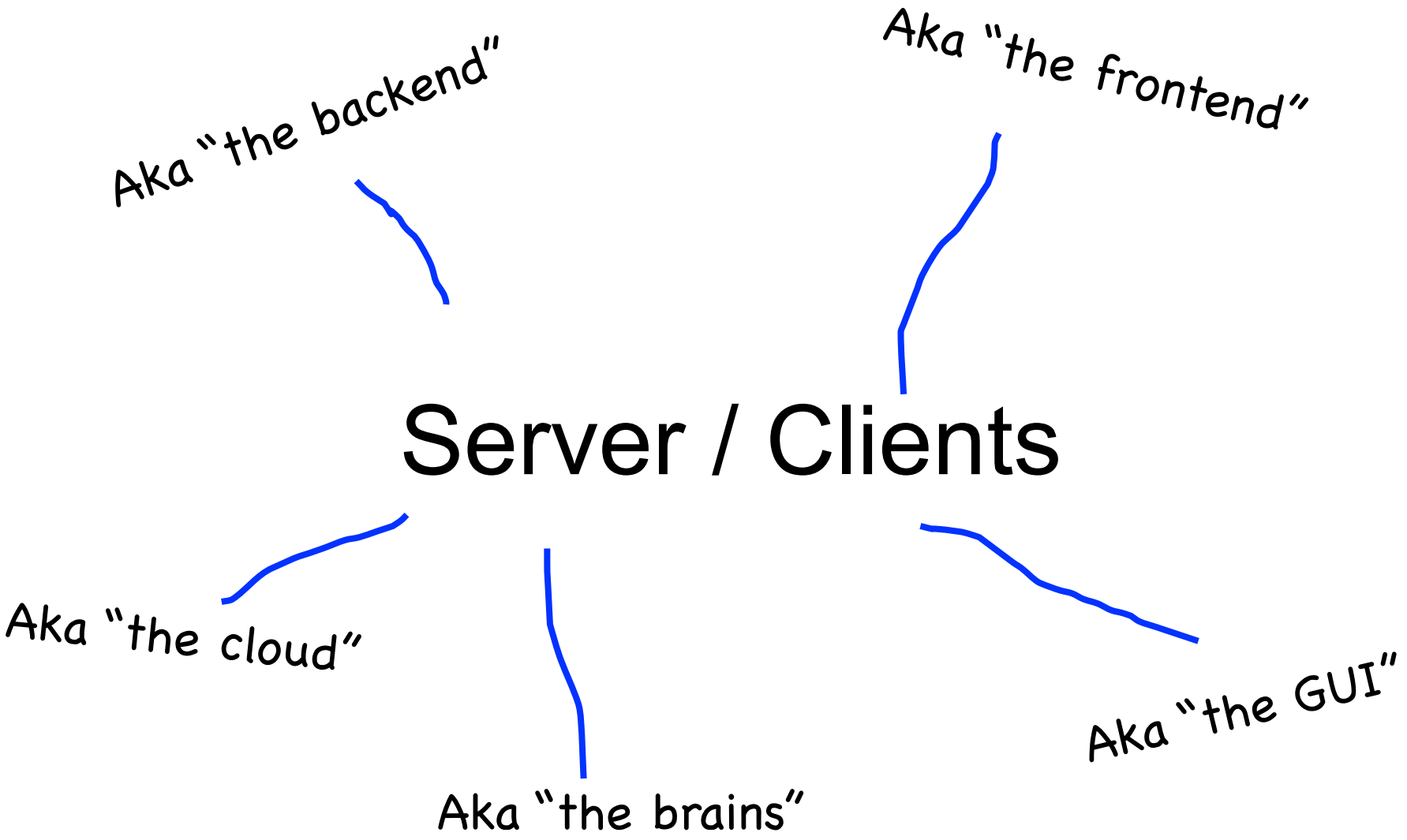


"Client"

Your mom's computer (linux shell)

"Client"

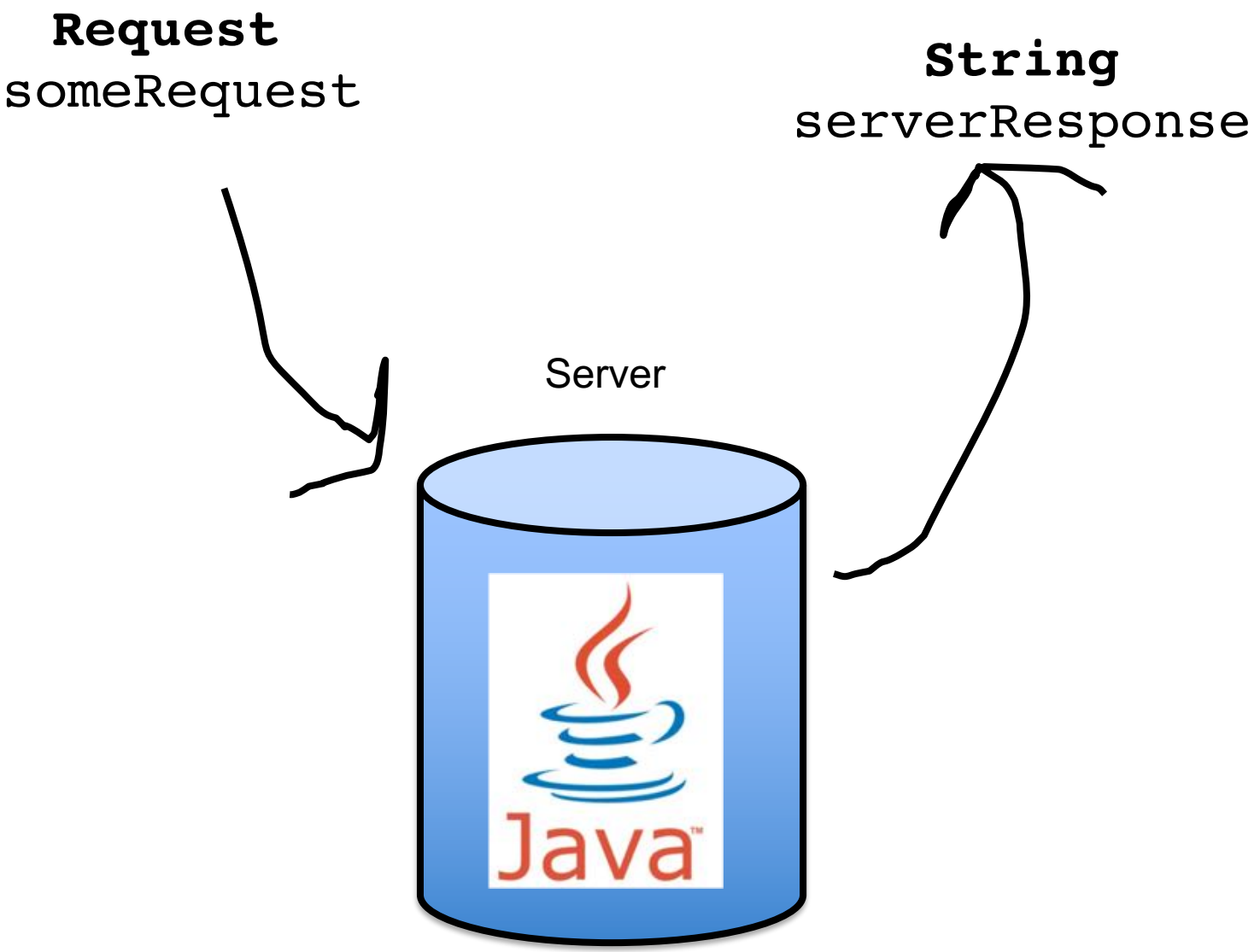
Most of the Internet



There are two types of internet programs. Servers and Clients

First, the server

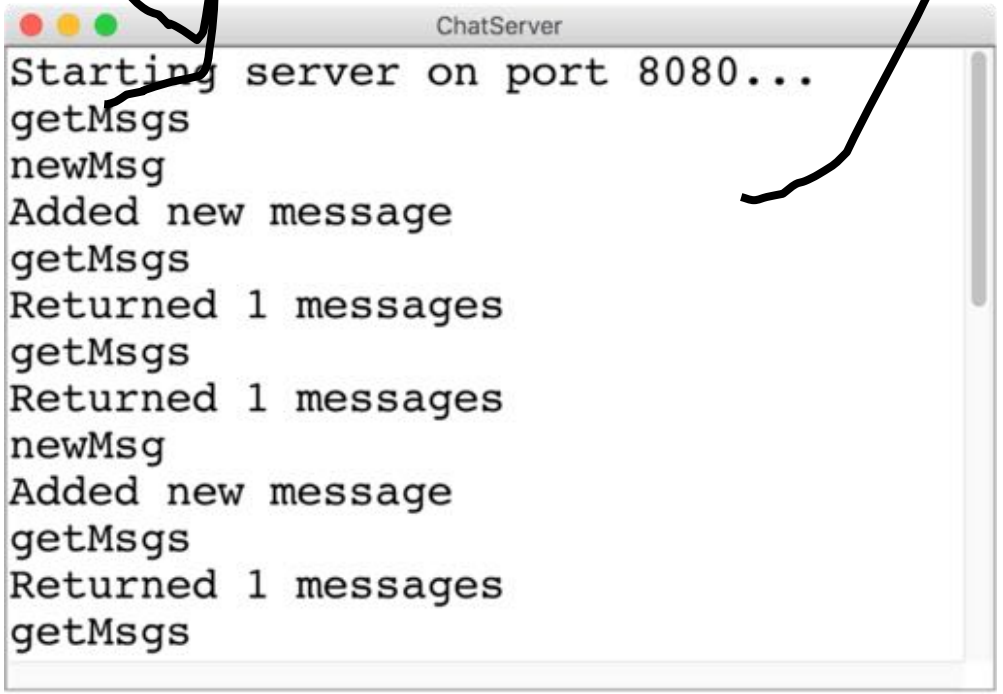
A Server's Simple Purpose



A Server's Simple Purpose

Request
`someRequest`

String
`serverResponse`



Servers on one slide

1

```
public String requestMade(Request request) {
    // server code goes here
}
```

2

```
// make a Server object
private SimpleServer server
      = new SimpleServer(this, 8000);
```

3

```
public void run(){
    // start the server
    server.start();
}
```


A Server's Simple Purpose

1

```
public String requestMade(Request request) {
    // server code goes here
}
```

2

```
// make a Server object
private SimpleServer server
      = new SimpleServer(this, 8000);
```

3

```
public void run(){
    // start the server
    server.start();
}
```

What is a Request?

```
/* Request has a command */
String command;

/* Request has parameters */
HashMap<String,String> params;
```

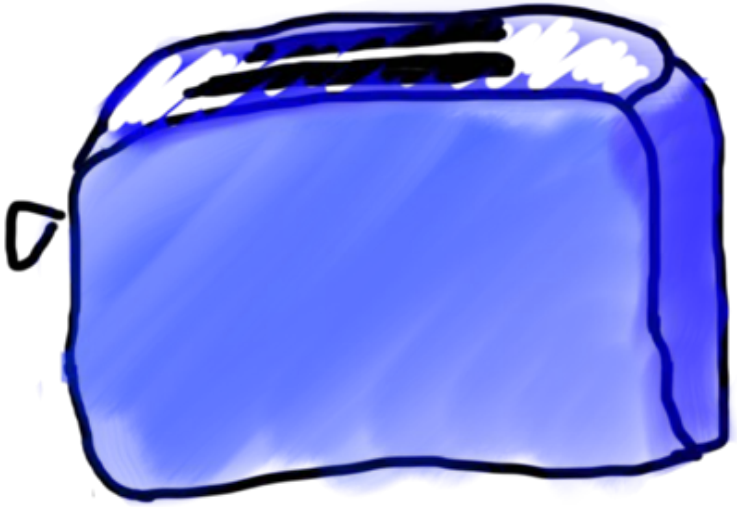
Request request

```
// methods that the server calls on requests
request.getCommand();
request.getParam(key); //returns associated value
```

Requests are like Remote Method Calls

Server has a bunch of discrete things it can do

Server

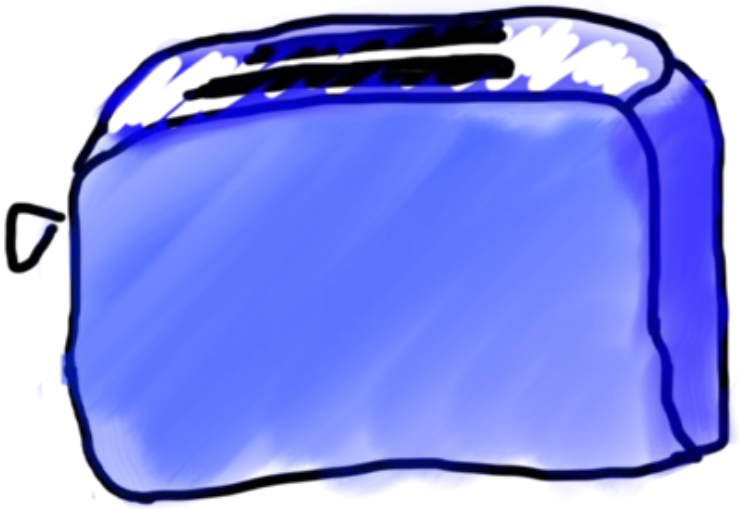


makeToast

getChocolate

Requests are like Remote Method Calls

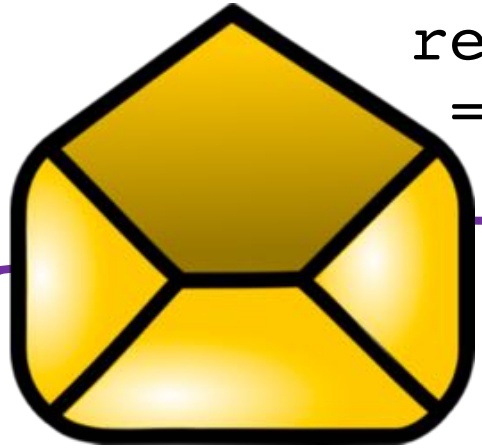
Server



makeToast

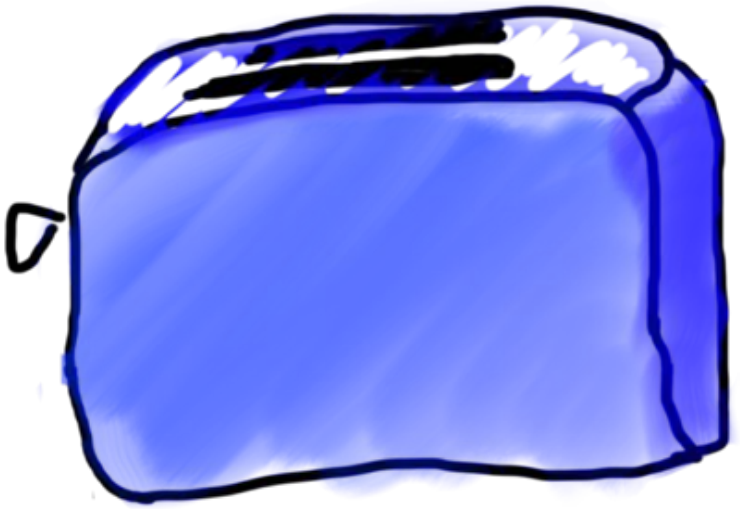
getChocolate

Requests are like Remote Method Calls



```
request.getCommand();
 => "makeToast"
```

Server

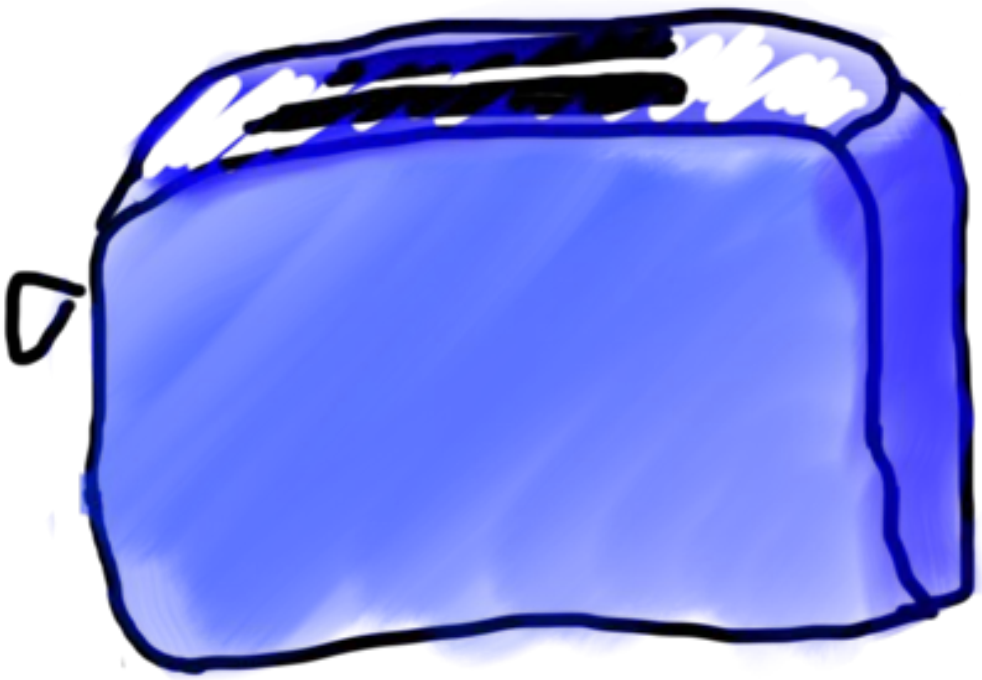


makeToast

getChocolate

Requests are like Remote Method Calls

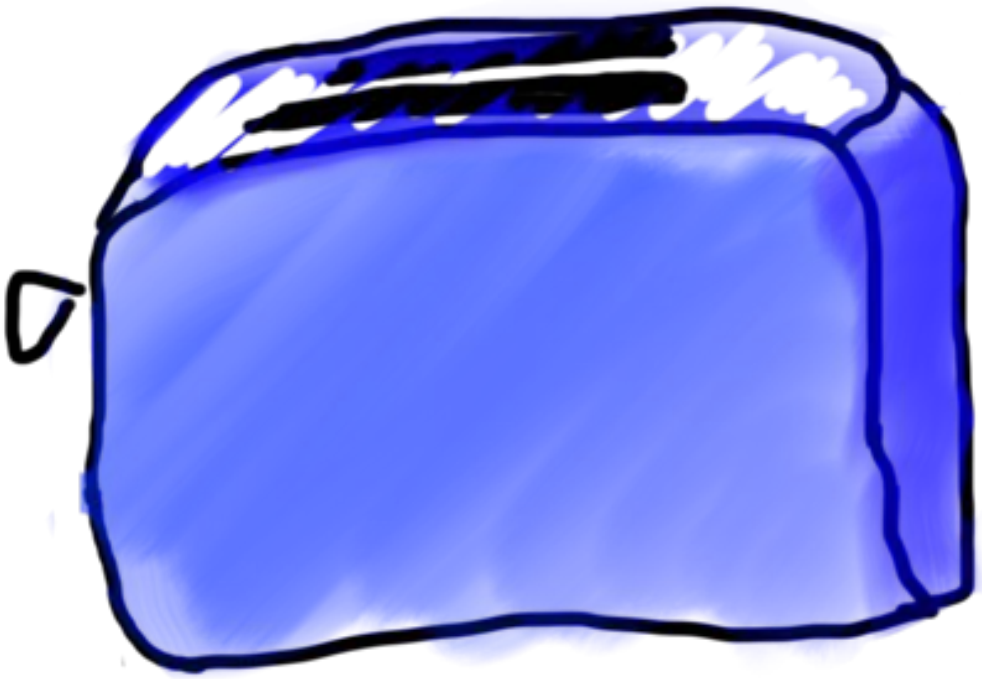
To make toast, I need a parameter which is the kind of bread



makeToast

Requests are like Remote Method Calls

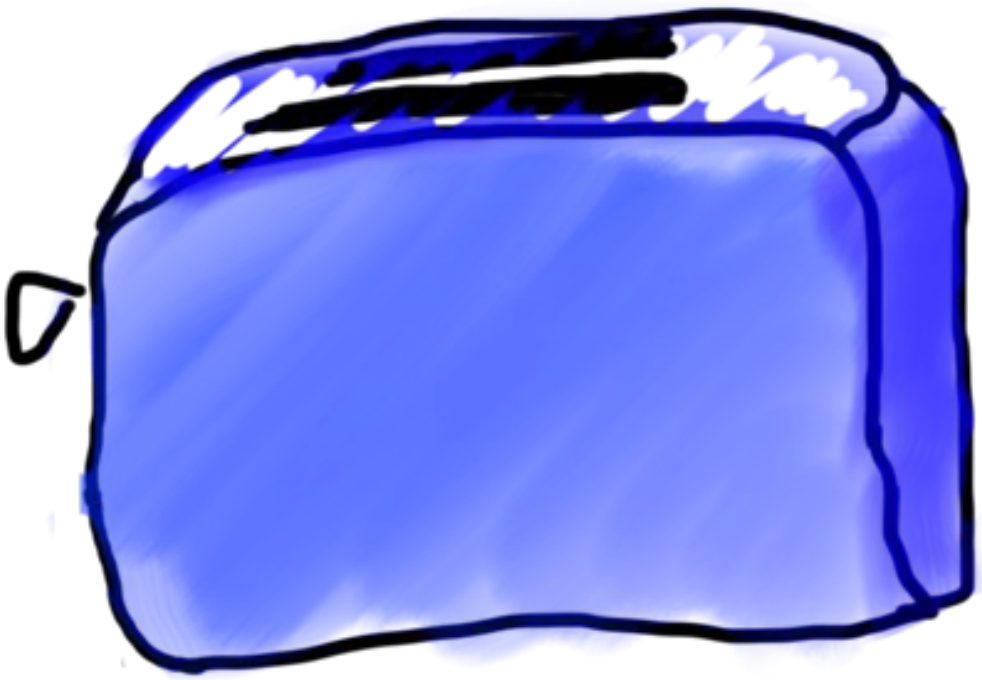
I was given a parameter!



makeToast

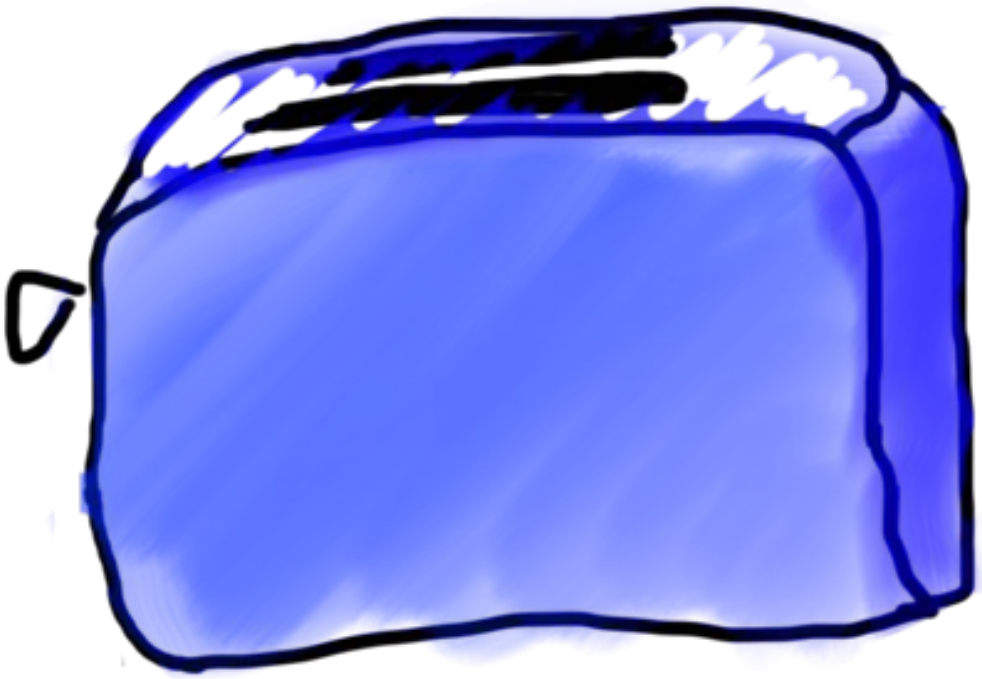
Requests are like Remote Method Calls

`request.getParam("bread")`



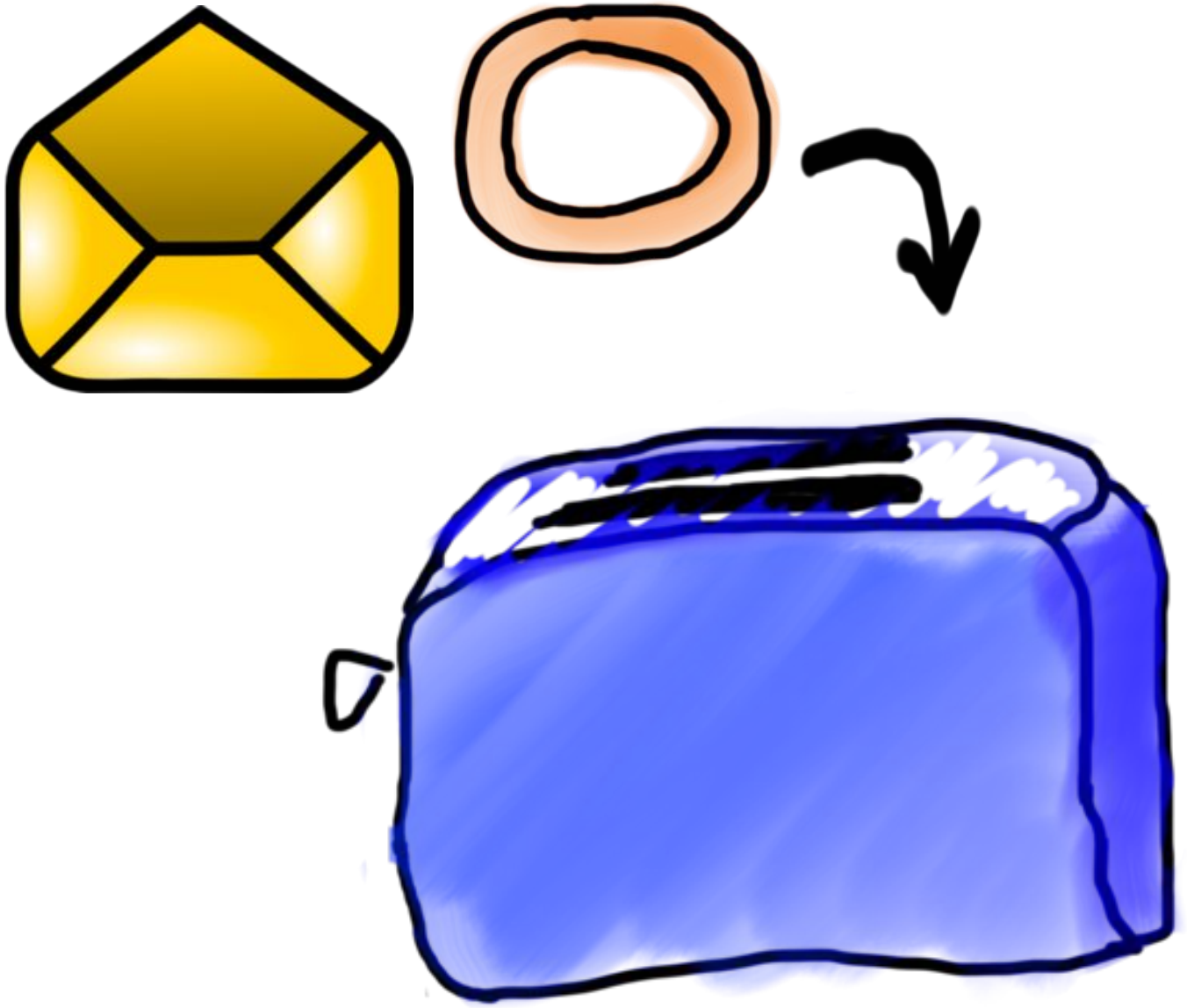
makeToast

Requests are like Remote Method Calls



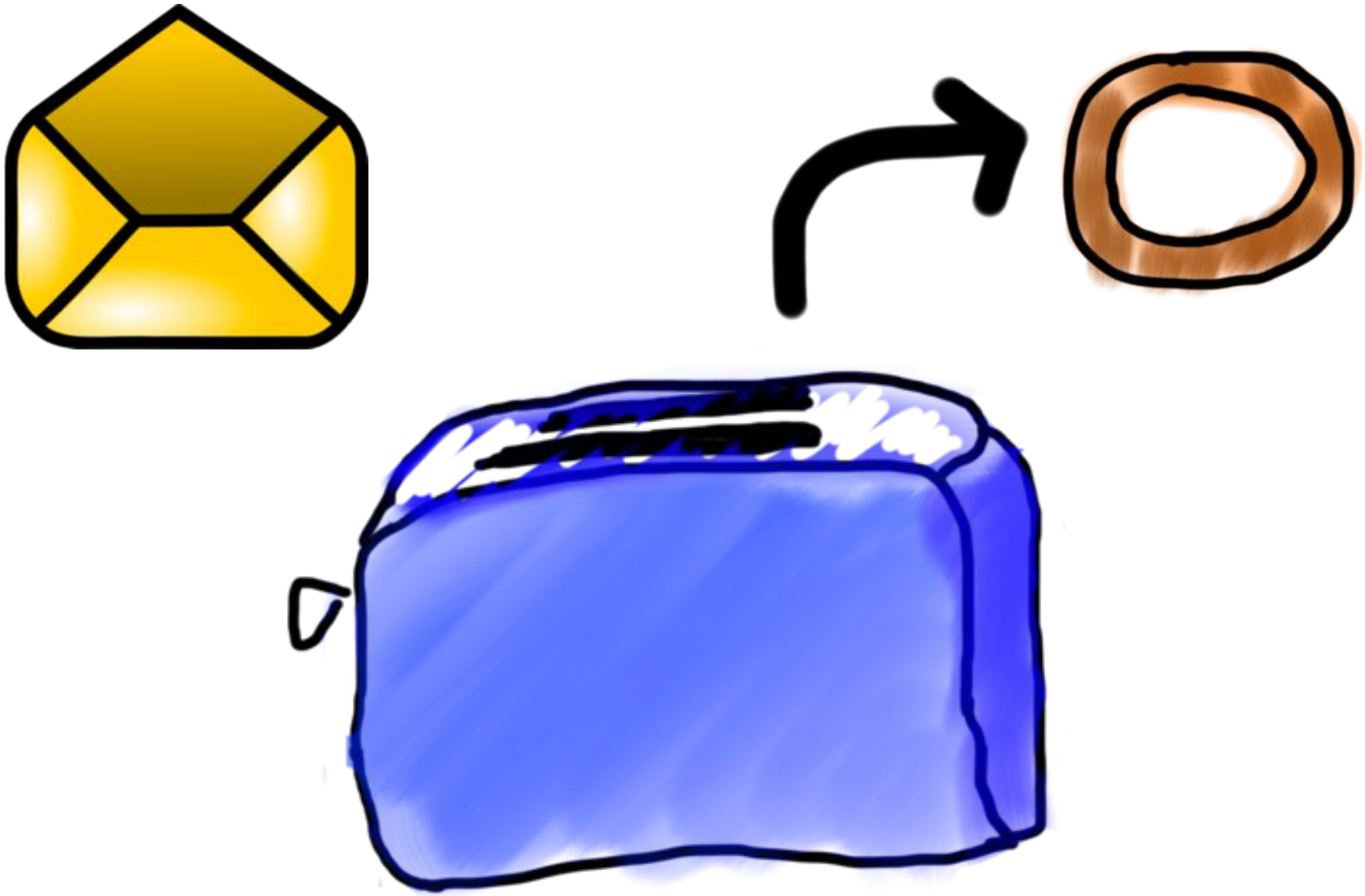
makeToast

Requests are like Remote Method Calls

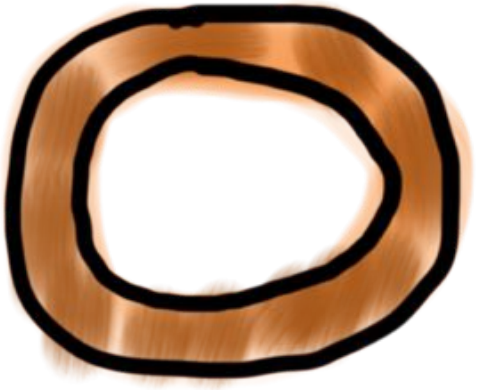


makeToast

Requests are like Remote Method Calls

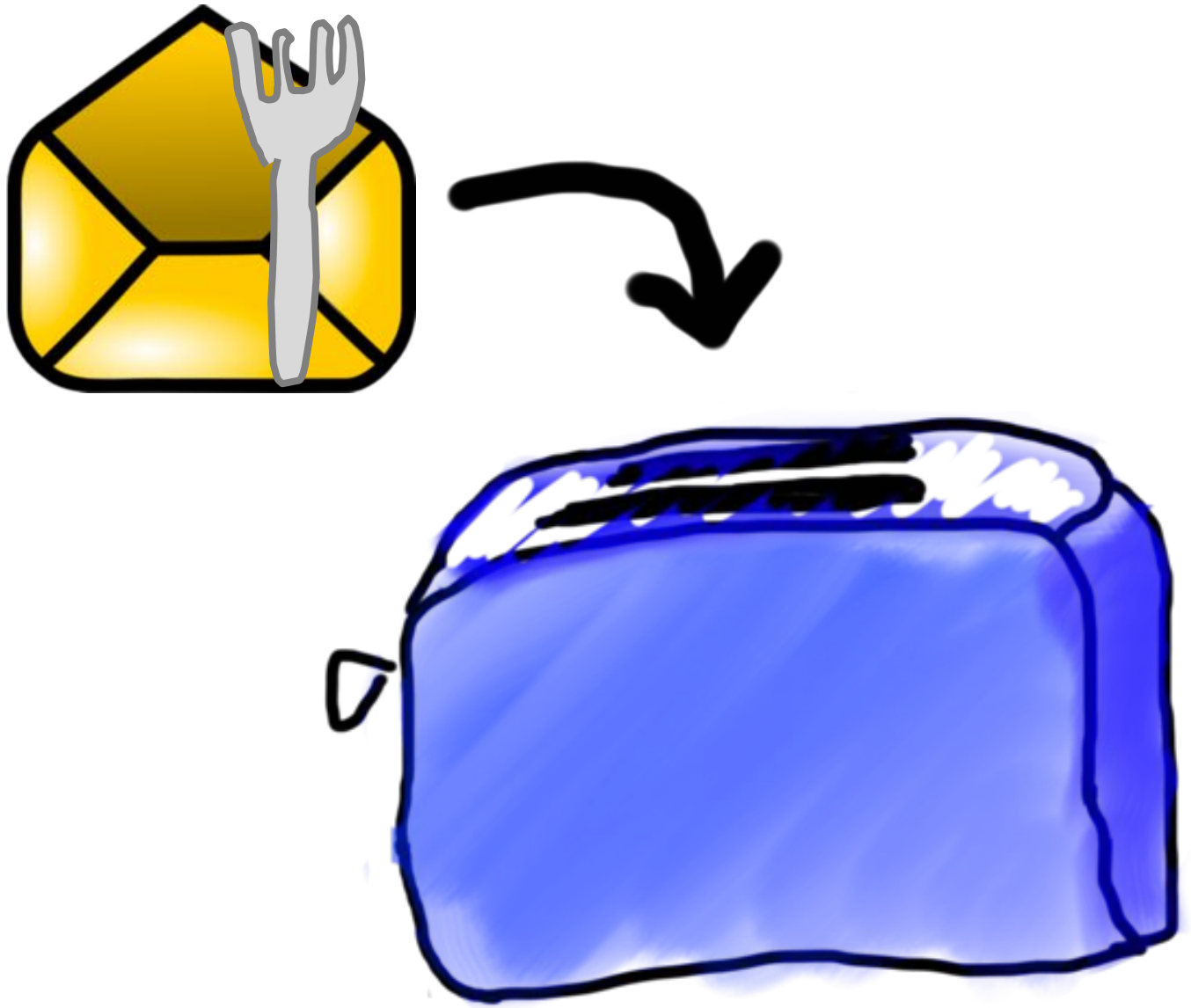


```
public String requestMade(Request request) {
    String cmd = request.getCommand();
    if(cmd.equals("makeToast")) {
        Bread input = request.getParam("bread");
        Bread output = runToaster(input);
        return output.toString();
    }
    ...
}
```



.toString()???

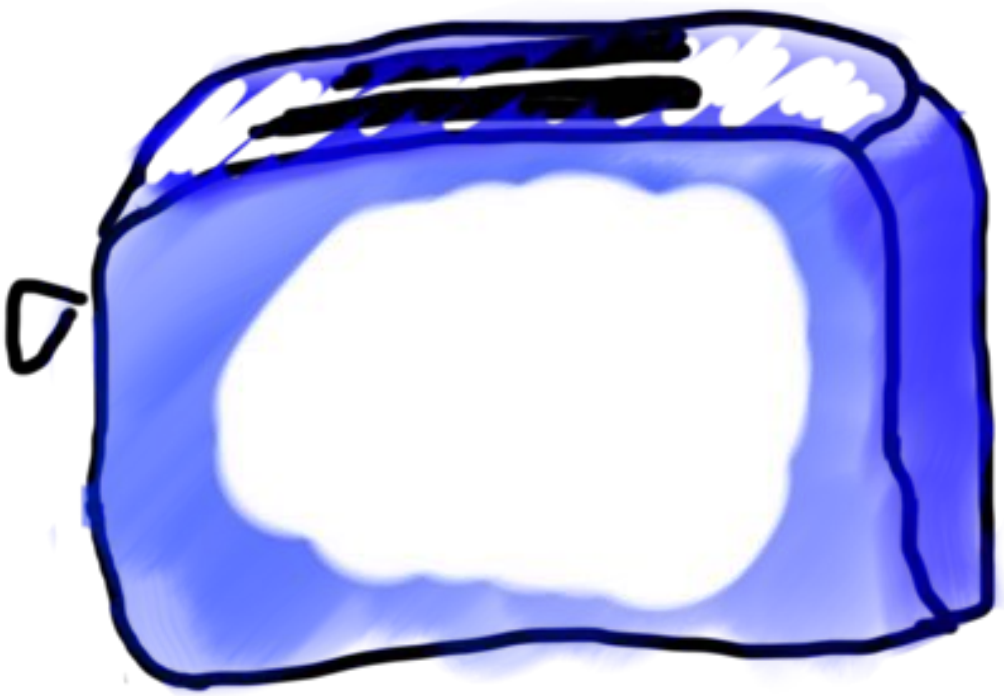
Requests are like Remote Method Calls



Requests are like Remote Method Calls

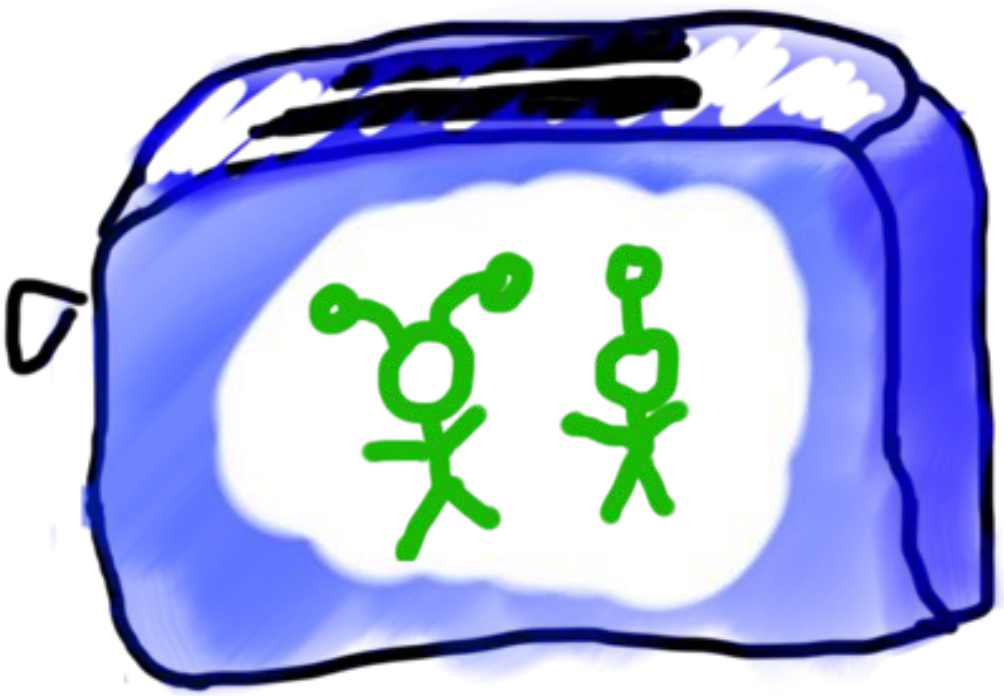
Requests are like Remote Method Calls

Requests are like Remote Method Calls



Requests are like Remote Method Calls

Requests are like Remote Method Calls



What is a Request?

```
/* Request has a command */
String command;

/* Request has parameters */
HashMap<String,String> params;
```

Request request

```
// methods that the server calls on requests
request.getCommand();
request.getParam(key); //returns associated value
```

A Server's Simple Purpose

1

```
public String requestMade(Request request) {
    // server code goes here
}
```

2

```
// make a Server object
private SimpleServer server
      = new SimpleServer(this, 8000);
```

3

```
public void run(){
    // start the server
    server.start();
}
```

A Server's Simple Purpose

1

```
public String requestMade(Request request) {
    // server code goes here
}
```

2

```
// make a Server object
private SimpleServer server
      = new SimpleServer(this, 8000);
```

3

```
public void run(){
    // start the server
    server.start();
}
```


What is a Port?

A Server's Simple Purpose

1

```
public String requestMade(Request request) {
    // server code goes here
}
```

2

```
// make a Server object
private SimpleServer server
      = new SimpleServer(this, 8000);
```

3

```
public void run(){
    // start the server
    server.start();
}
```


Servers on one slide

1

```
public String requestMade(Request request) {
    // server code goes here
}
```

2

```
// make a Server object
private SimpleServer server
      = new SimpleServer(this, 8000);
```

3

```
public void run(){
    // start the server
    server.start();
}
```


Echo Server


localhost:8090/hellocs106a
Chris Piech
localhost:8090/hellocs106a
Your command was 11 chars long.

Echo Server

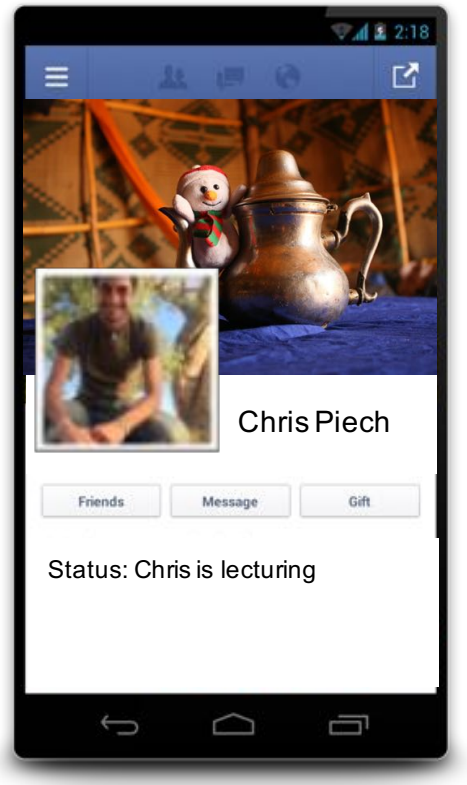
Request
Any Request

String
Length of the cmd

There are two types of internet programs. Servers and Clients

Then, the client

A Clients's Purpose



1. Interact with the user

2. Get data from its server

3. Save data to its server

Clients on one slide

```
try {

  // 1. construct a new request
  Request example = new Request("getStatus");

  // 2. add parameters to the request
  example.addParam("name", "chris");

  // 3. send the request to a computer on the internet
  String result = SimpleClient.makeRequest(HOST, example);

} catch(IOException e) {

  // The internet is a fast and wild world my friend

}
```


Clients on one slide

```
try {

  // 1. construct a new request
  Request example = new Request("getStatus");

  // 2. add parameters to the request
  example.addParam("name", "chris");

  // 3. send the request to a computer on the internet
  String result = SimpleClient.makeRequest(HOST, example);

} catch(IOException e) {

  // The internet is a fast and wild world my friend

}
```


Clients on one slide

```
try {

  // 1. construct a new request
  Request example = new Request("getStatus");

  // 2. add parameters to the request
  example.addParam("name", "chris");

  // 3. send the request to a computer on the internet
  String result = SimpleClient.makeRequest(HOST, example);

} catch(IOException e) {

  // The internet is a fast and wild world my friend

}
```

Clients on one slide

```
try {

  // 1. construct a new request
  Request example = new Request("getStatus");

  // 2. add parameters to the request
  example.addParam("name", "chris");

  // 3. send the request to a computer on the internet
  String result = SimpleClient.makeRequest(HOST, example);

} catch(IOException e) {

  // The internet is a fast and wild world my friend

}
```

Clients on one slide

```
try {

  // 1. construct a new request
  Request example = new Request("getStatus");

  // 2. add parameters to the request
  example.addParam("name", "chris");

  // 3. send the request to a computer on the internet
  String result = SimpleClient.makeRequest(HOST, example);

} catch(IOException e) {

  // The internet is a fast and wild world my friend

}
```


Clients on one slide

```
try {

  // 1. construct a new request
  Request example = new Request("getStatus");

  // 2. add parameters to the request
  example.addParam("name", "chris");

  // 3. send the request to a computer on the internet
  String result = SimpleClient.makeRequest(HOST, example);

} catch(IOException e) {

  // The internet is a fast and wild world my friend

}
```


Clients on one slide

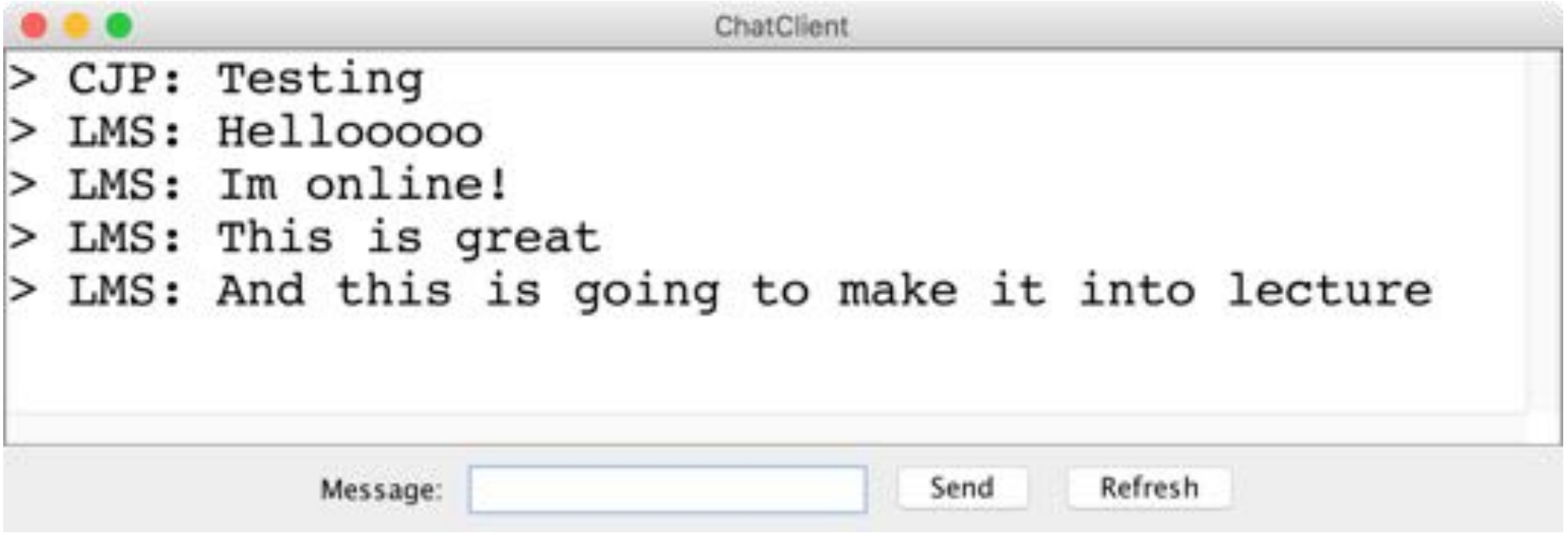
```
try {

  // 1. construct a new request
  Request example = new Request("getStatus");

  // 2. add parameters to the request
  example.addParam("name", "chris");

  // 3. send the request to a computer on the internet
  String result = SimpleClient.makeRequest(HOST, example);

} catch(IOException e) {

  // The internet is a fast and wild world my friend

}
```


Clients on one slide

```
try {

  // 1. construct a new request
  Request example = new Request("getStatus");

  // 2. add parameters to the request
  example.addParam("name", "chris");

  // 3. send the request to a computer on the internet
  String result = SimpleClient.makeRequest(HOST, example);

} catch(IOException e) {

  // The internet is a fast and wild world my friend

}
```

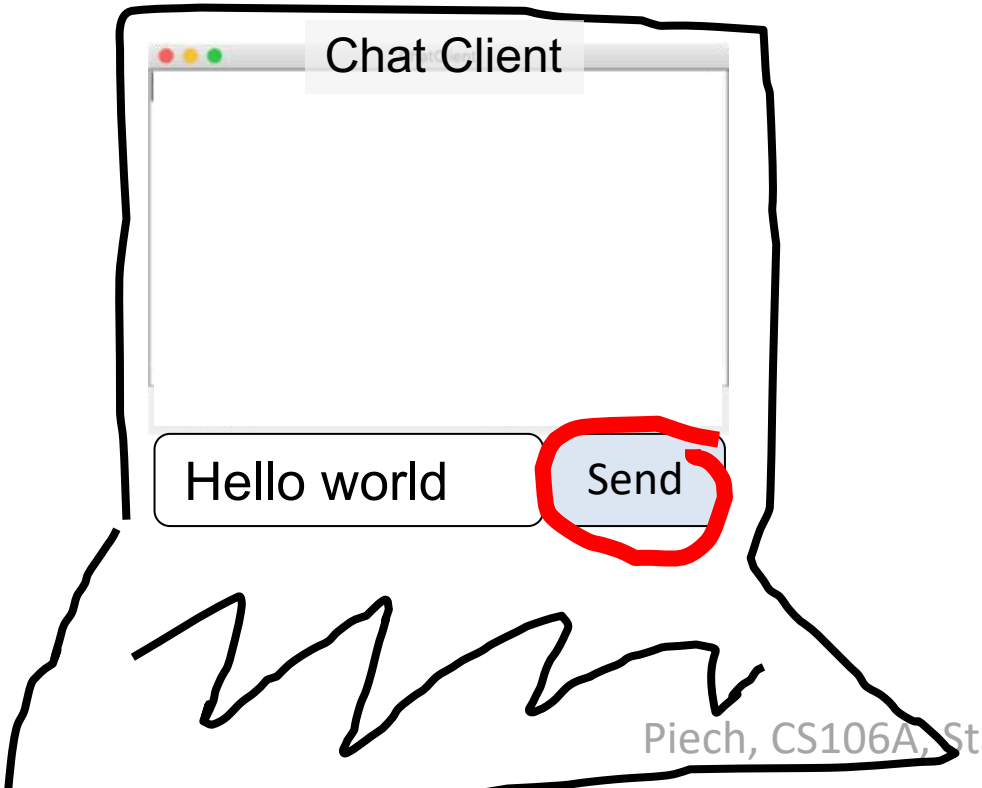
Time for a little chat

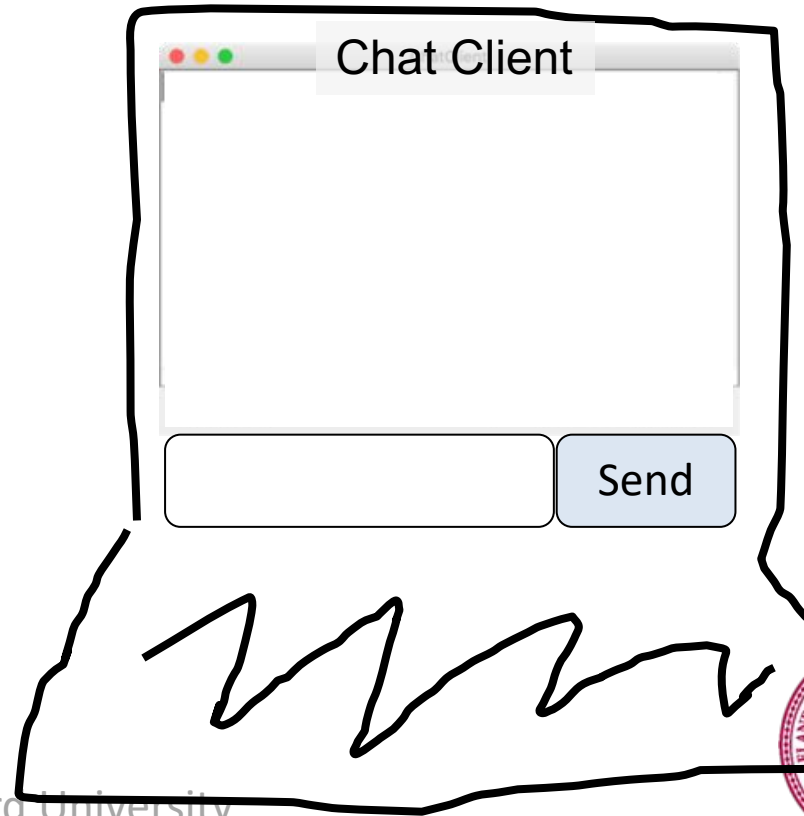
Chat Server and Client




```
history = [
]
```

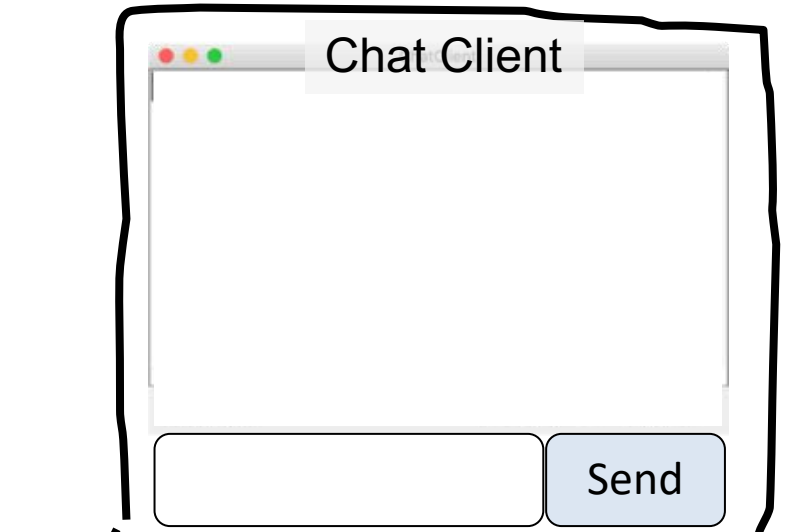
```
addMsg
msg = Hello world
```

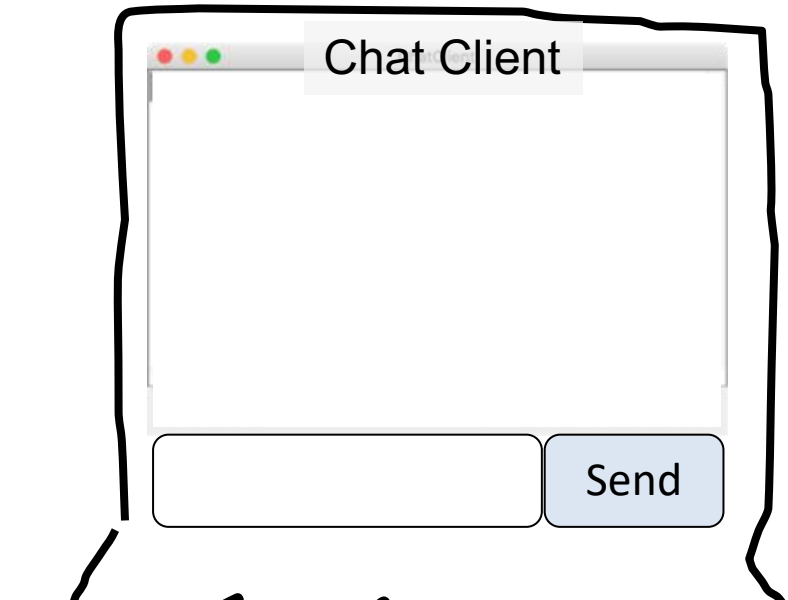





```
history = [
    Hello world
]
```

```
getMsgs
index = 0
```



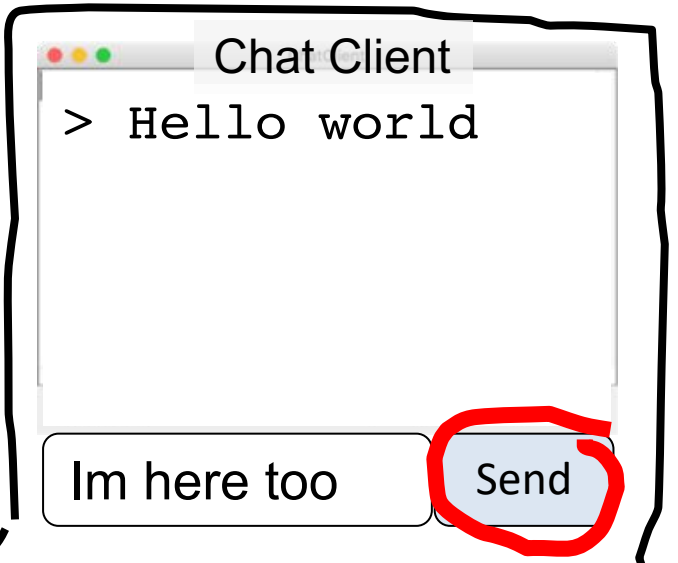



```
history = [
    Hello world
]
```

```
[Hello world]
```

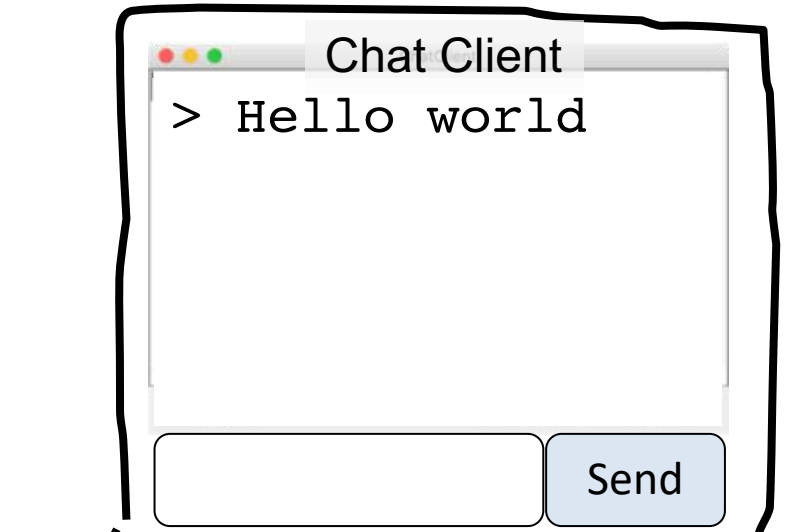

```
history = [
    Hello world
]
```

```
addMsg
msg = Im here too
```




```
history = [
    Hello world,
    Im here too
]
```

```
getMsgs
index = 1
```




```
history = [
    Hello world,
    Im here too
]
```

[Im here too]

Chat Server

Chat Server

addMsg
msg = *text*

getMsgs
index = *startIndex*