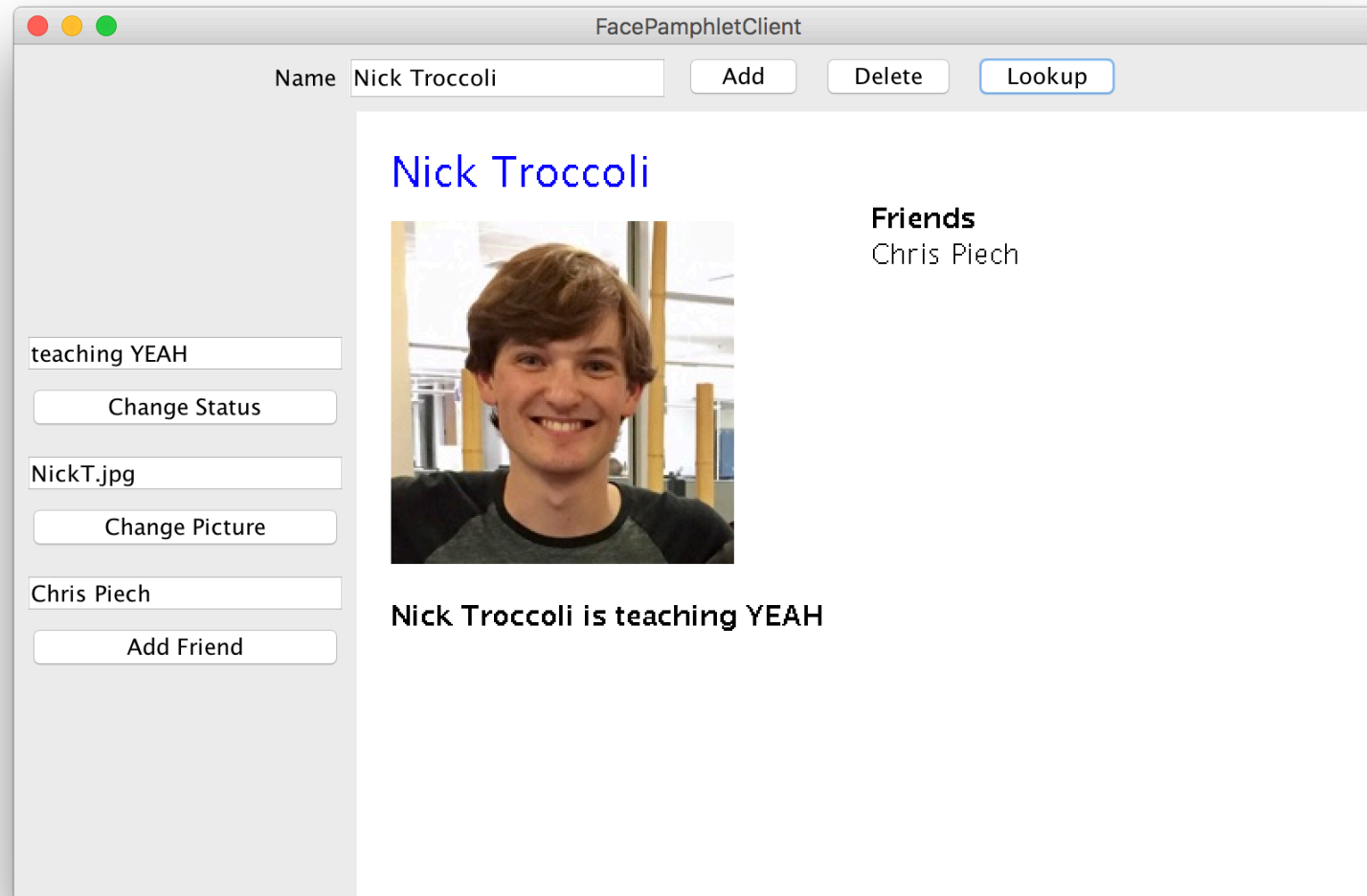


YEAH 7: FacePamphlet

3/9/17, 7-8PM – Nick Troccoli



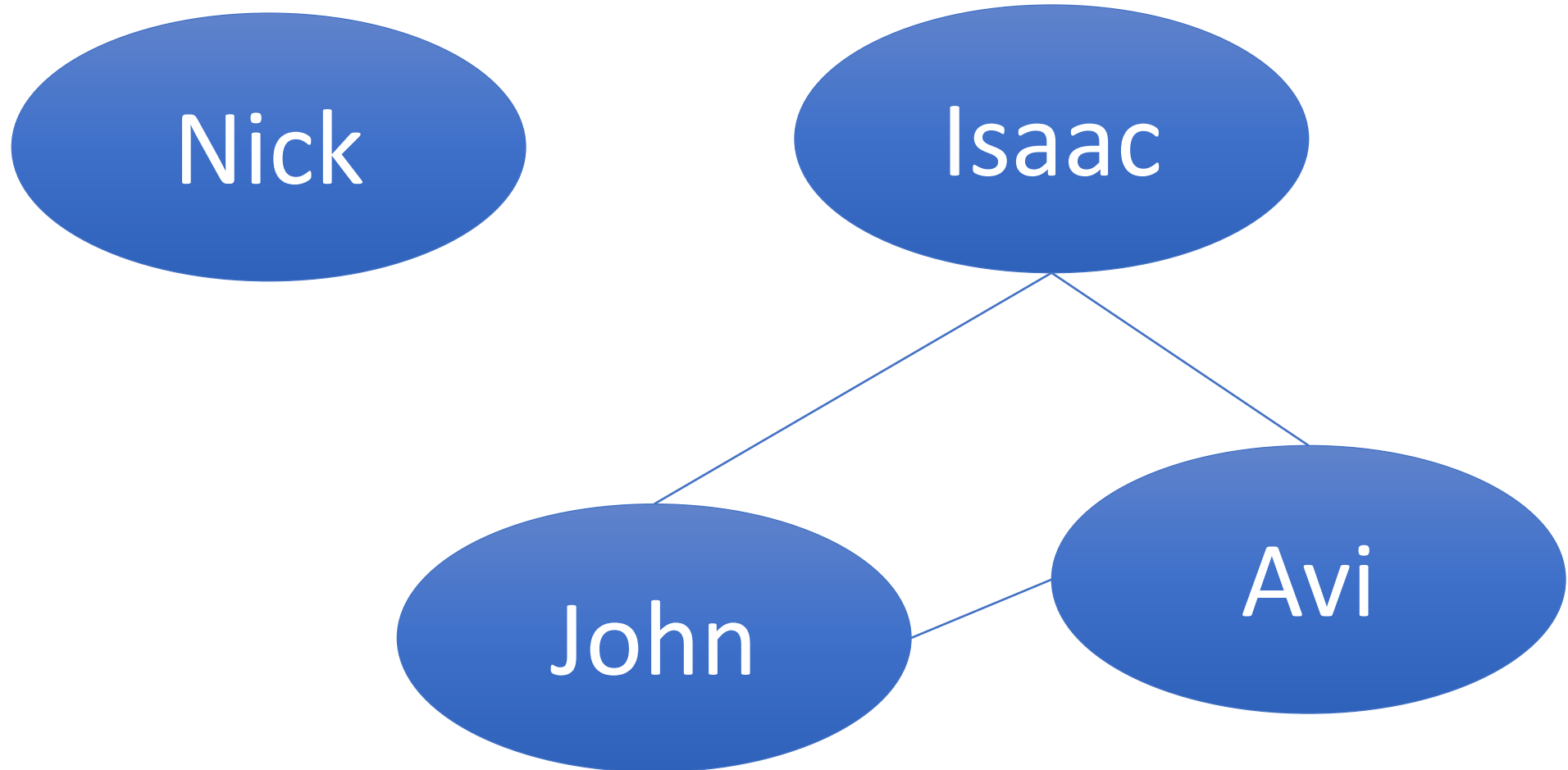
Outline for YEAH

- Overview of social networks
- Overview of internet programs
- Assignment demo
- **FacePamphletServer** walkthrough
- **FacePamphletClient** walkthrough

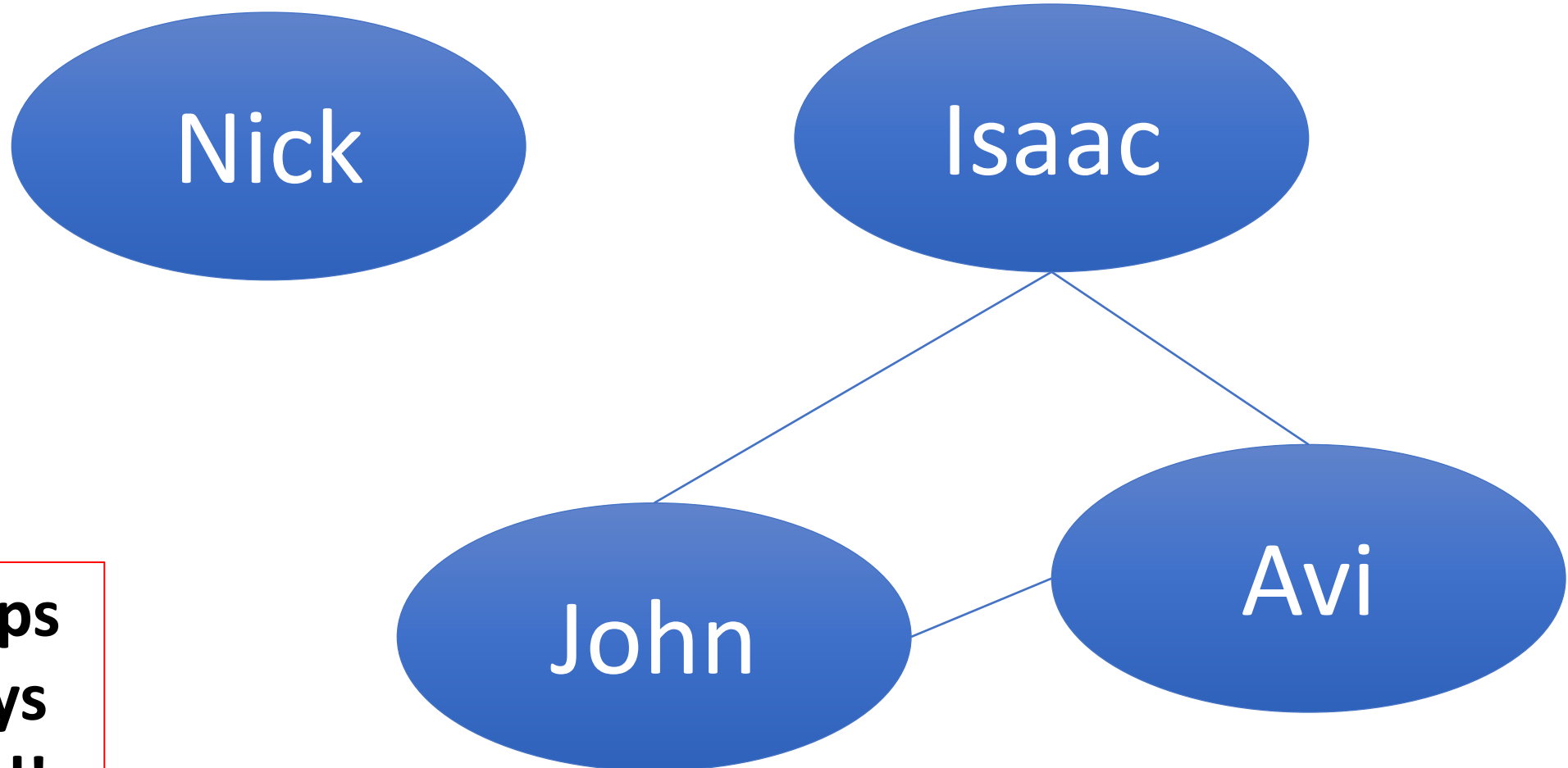
Outline for YEAH

- Overview of social networks
- Overview of internet programs
- Assignment demo
- **FacePamphletServer** walkthrough
- **FacePamphletClient** walkthrough

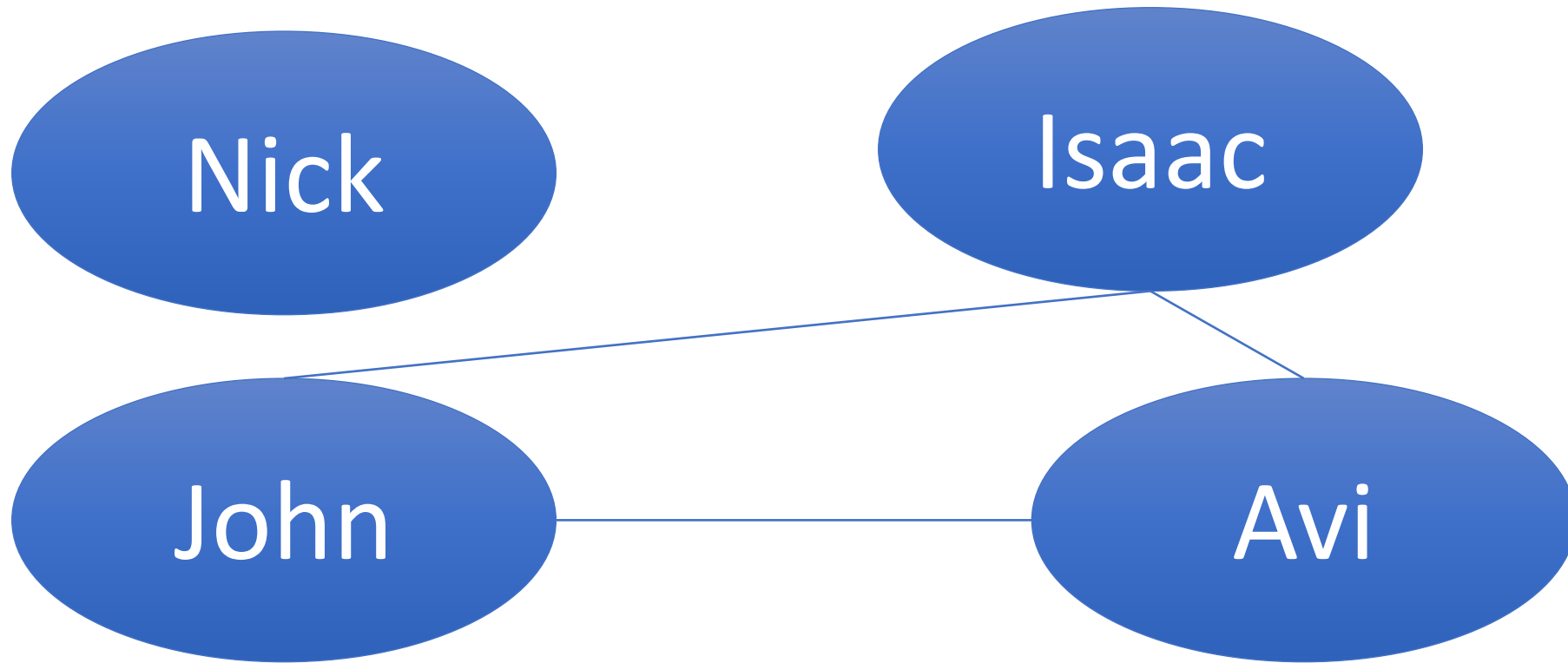
What is a Social Network?



What is a Social Network?



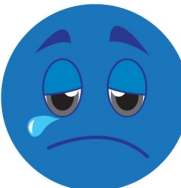
**Friendships
are always
reciprocal!**



Isaac's
Friends:
John
Avi

John's
Friends:
Isaac
Avi

Avi's
Friends:
John
Isaac

Nick's
Friends:


FacePamphlet!

- Due at 10:30AM on Friday, March 17
- **NO LATE DAYS OR LATE SUBMISSIONS!**
- Practice with collections, classes, and internet programs (!!)

Outline for YEAH

- Overview of social networks
- Overview of internet programs
- Assignment demo
- **FacePamphletServer** walkthrough
- **FacePamphletClient** walkthrough

Some background: the Internet



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

Some background: the Internet



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

Some background: the Internet



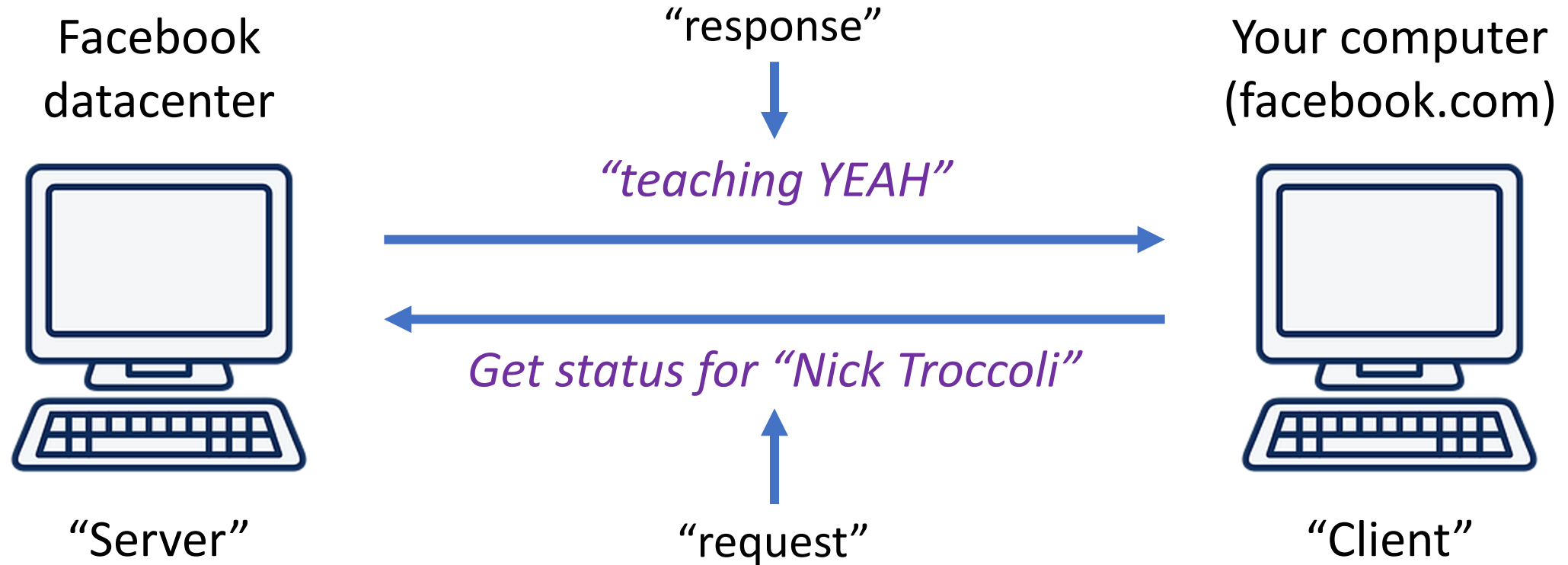
The internet is just many programs sending messages (as *Strings*)

Some background: the Internet



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

Some background: the Internet



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

FacePamphlet

FacePamphletServer



“Server”



Your task

FacePamphletClient



“Client”



Extra credit

RESPONSE



REQUEST



Outline for YEAH


- Overview of social networks
- Overview of internet programs
- **Assignment demo**
- **FacePamphletServer** walkthrough
- **FacePamphletClient** walkthrough

DEMO

FacePamphletClient

Name

Nick Troccoli



Friends
Chris Piech

Nick Troccoli is teaching YEAH

Outline for YEAH

- Overview of social networks
- Overview of internet programs
- Assignment demo
- **FacePamphletServer** walkthrough
- **FacePamphletClient** walkthrough

What Comes In The Box?

- **FacePamphletServer** – your server program
- **FacePamphletProfile** – a class that represents a single user's profile
- **ServerTester** – tests we provide to help you test your server
- **FacePamphletClient.jar** – fully-functional FacePamphlet client

What Comes In The Box?

- **FacePamphletServer** – your server program
- **FacePamphletProfile** – a class that represents a single user's profile
- **ServerTester** – tests we provide to help you test your server
- **FacePamphletClient.jar** – fully-functional FacePamphlet client

How does the server program work?


```
/**  
 * Starts the server running so that when a program sends  
 * a request to this computer, the method requestMade is  
 * called.  
 */
```

```
public void run() {  
    println("Starting server on port " + PORT);  
    server.start();  
}
```

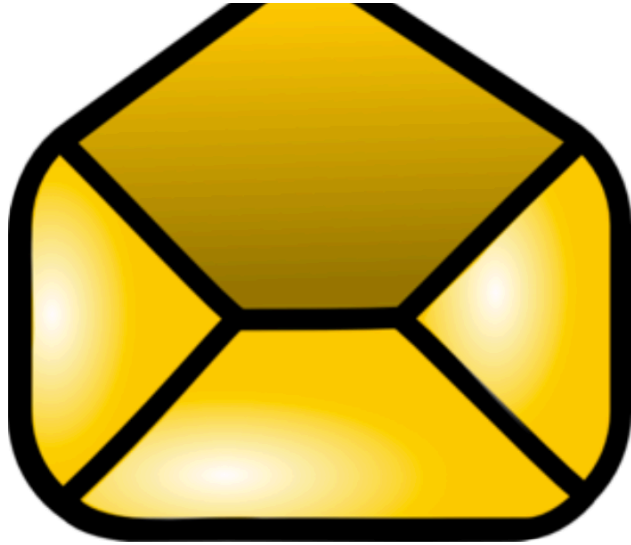
```
/**  
 * When a request is sent to this computer, this method is  
 * called. It must return a String.  
 */
```

```
public String requestMade(Request request) {  
    String cmd = request.getCommand();  
    println(request.toString());  
  
    // your code here.  
  
    return "Error: Unknown command " + cmd + ".";  
}
```

Respond to requests here. *The String you return will be sent as the response.*



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
```

So what requests should our server handle?

- addProfile (**name** parameter)
- containsProfile (**name** parameter)
- deleteProfile (**name** parameter)
- getStatus (**name** parameter)
- getImgFileName (**name** parameter)
- getFriends (**name** parameter)
- setStatus (**name** and **status** parameters)
- setImgFileName (**name** and **fileName** parameters)
- addFriend (**name1** and **name2** parameters)

What Comes In The Box?

- **FacePamphletServer** – your server program
- **FacePamphletProfile** – a class that represents a single user's profile
- **ServerTester** – tests we provide to help you test your server
- **FacePamphletClient.jar** – fully-functional FacePamphlet client

FacePamphletProfile

- A new variable type that represents one user's profile.
- Includes:
 - Name (given when a profile is created)
 - Status (initially empty string)
 - Image filename (initially empty string)
 - List of friend names (???)

Just like in NameSurfer, we provide a “stub” – you must implement!

What Comes In The Box?

- **FacePamphletServer** – your server program
- **FacePamphletProfile** – a class that represents a single user's profile
- **ServerTester** – tests we provide to help you test your server
- **FacePamphletClient.jar** – fully-functional FacePamphlet client

Milestone 1: ping

- Write a little code to make your server respond to the **ping** command
- Your server should respond with “hello, internet”

Milestone 2: FacePamphletProfile

- Implement the **FacePamphletProfile** class
- We provide “stub” methods; you must fill these in
- **Do not change the public methods at all!**
- Your **FacePamphletServer** will have to be able to look up a user’s profile based on their name – what data structure can you use?
 - **HINT:** think NameSurfer

Milestone 3: Handling your first requests

- **addProfile** (**name** parameter): creates a profile with the given name. Returns “success” or, if the profile already exists, returns an error message.
- **containsProfile** (**name** parameter): checks if a profile with the given name exists. Returns “true” or “false” (*as Strings!!*)
- **deleteProfile** (**name** parameter): Removes a profile from the server’s database. Returns “success” or, if the profile doesn’t exist, returns an error message.
- All error messages should start with “Error: ”.

Milestone 3: Handling your first requests

- **deleteProfile** (**name** parameter): Removes a profile from the server's database. Returns "success" or, if the profile doesn't exist, returns an error message.

Remember: if you delete a profile, you must also remove that person from all of their friends' friend lists!

Milestone 3: Handling your first requests

- **addProfile** (**name** parameter): creates a profile with the given name. Returns “success” or, if the profile already exists, returns an error message.
- **containsProfile** (**name** parameter): checks if a profile with the given name exists. Returns “true” or “false” (*as Strings!!*)
- **deleteProfile** (**name** parameter): Removes a profile from the server’s database. Returns “success” or, if the profile doesn’t exist, returns an error message.

```
request.getParam("name"); // name associated with this request.
```

Milestone 4: Handling more requests

- **getStatus** (**name** parameter): returns the status of the user with the given name. Returns an error message if the profile doesn't exist.
- **setStatus** (**name** and **status** parameters): Sets the status of the user with the given name. Returns "success" or, if the profile doesn't exist, returns an error message.
- **getImgFileName** (**name** parameter): Returns the image filename of the user with the given name. Returns an error message if the profile doesn't exist.
- **setImgFileName** (**name** and **fileName** parameters): sets the image filename for the user with the given name. Returns "success" or, if the profile doesn't exist, returns an error message.

Milestone 5: Even more requests!

- **getFriends** (**name** parameter): returns the list of friends, *as a string*, for the user with the given name. Returns an error message if the profile doesn't exist.
- **addFriend** (**name1** and **name2** parameters): Makes the user with name1 friends with the user with name2, *and vice versa*. Returns "success", or an error message if:
 - **Either user does not exist**
 - **The users are already friends**
 - **They are the same person 😊**

Milestone 5: Even more requests!

- **getFriends** (**name** parameter): returns the list of friends, *as a string*, for the user with the given name. Returns an error message if the profile doesn't exist.

Must send back a string with the format:

```
"[Nick, Chris, Mehran]"
```

```
"[]" // if no friends
```

Hint: `ArrayList`'s `toString` method returns data in this format...

Tips

- **Remember:** everything sent back must be a string!
- **Use your browser to test (note: can only send 1-word params)**
 - <http://localhost:8000/addFriend?name1=Nick&name2=Chris>
 - <http://localhost:8000/setStatus?name=Nick&status=teaching>
 - <http://localhost:8000/addProfile?name=Nick>



Server location

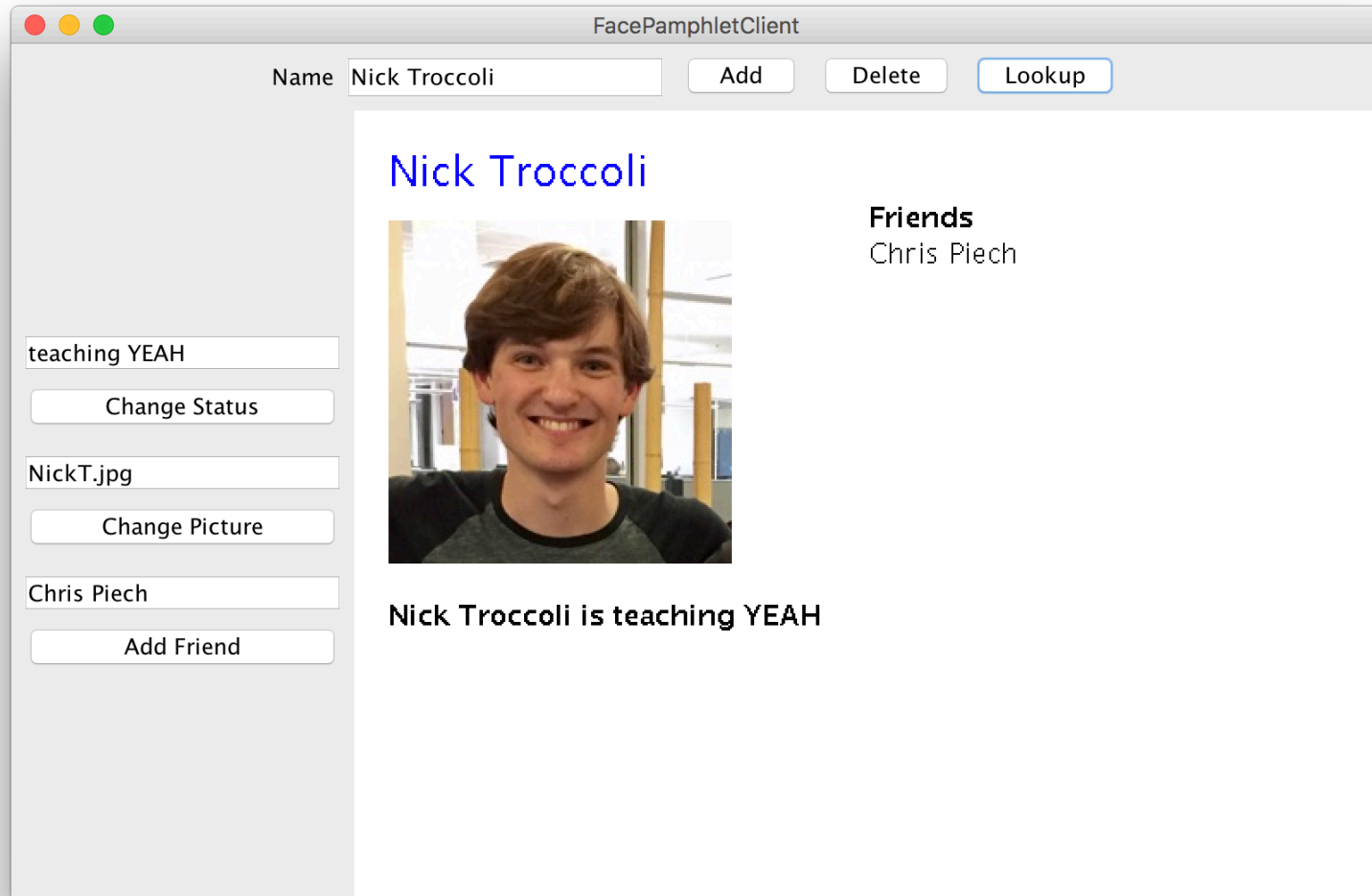
Request
command

Request
param(s)

Outline for YEAH

- Overview of social networks
- Overview of internet programs
- Assignment demo
- **FacePamphletServer** walkthrough
- **FacePamphletClient** walkthrough

Extra Credit: FacePamphletClient



Extra Credit: FacePamphletClient

- Implement the client program that we provide
- Practice with interactors, graphics, and sending requests to your server
- See the starter code and the **Chat** lecture example for how to send a request to your server.

Closing Tips

- Think about similarities between **FacePamphlet** and **NameSurfer**
- Go to the LaIR!
- Incorporate IG feedback
- Add awesome extensions!
- **Have fun!**