EECE1012
Net-Centric full-fledged to Computing
fully-fledged project

Amirhossein Chinaei, Winter 2019

Office Hours: T 9:30-10:30 W 11:30-12:30 LAS3048

ahchinaei@cse.yorku.ca

review: principle of layering

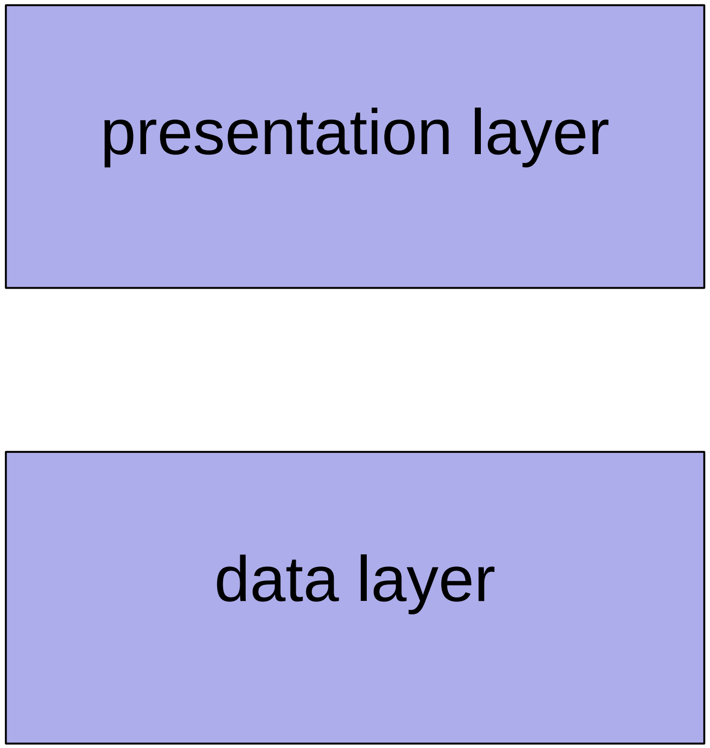
- dividing the application to two+ groups of classes
  - that are functionally or logically related
- such that each layer demonstrates cohesion
- and the dependency among classes is minimized

- **advantages:**
  - modularity, maintainability, reusability

- **disadvantages:**
  - reduced performance

review: 2-layer architecture

- simple application functionality

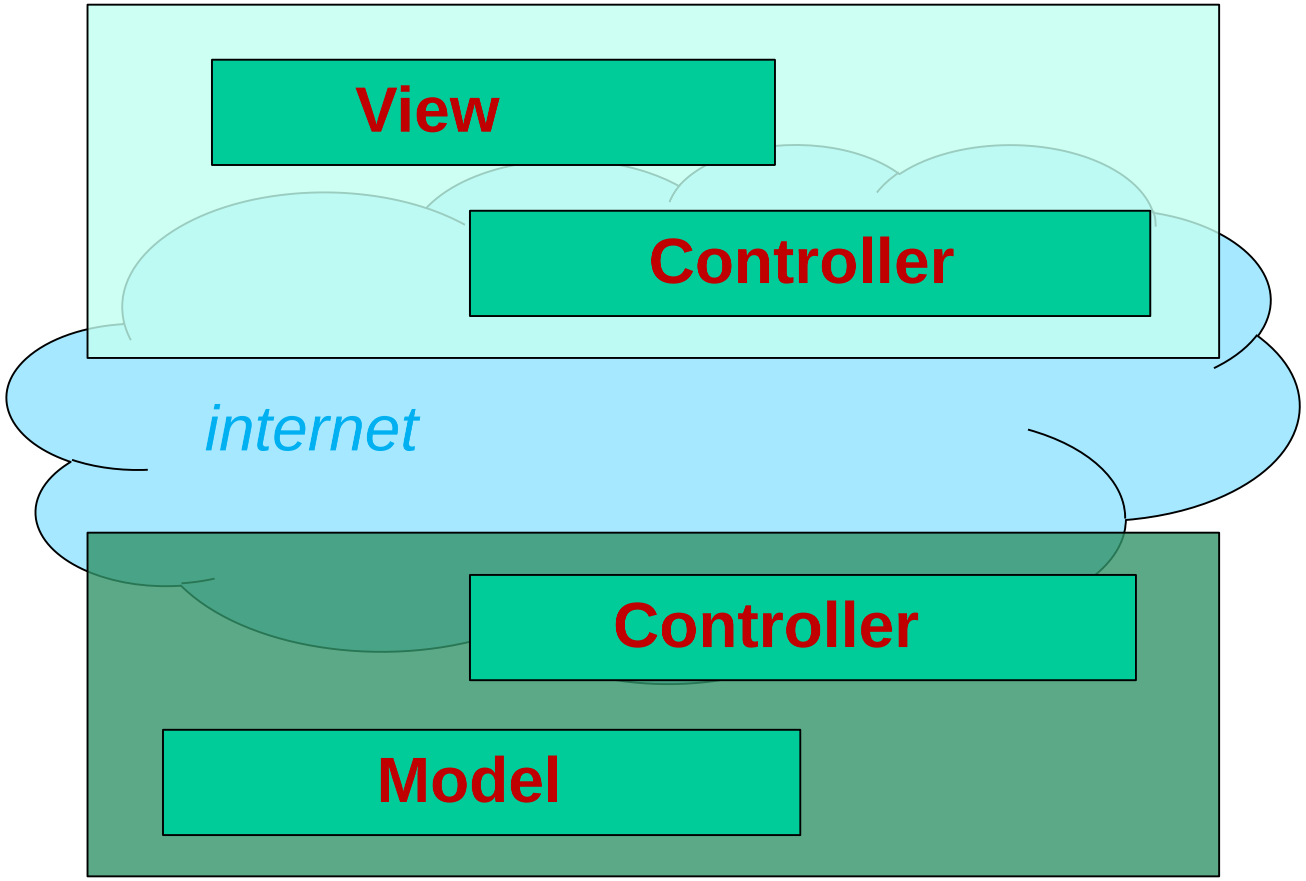


review: mvc

- the **model** tier
  - represents the **data and logic**
- the **view** tier
  - represents the **user interface**
- the **controller** tier
  - connects and coordinates—**controls**—activities between the view and model

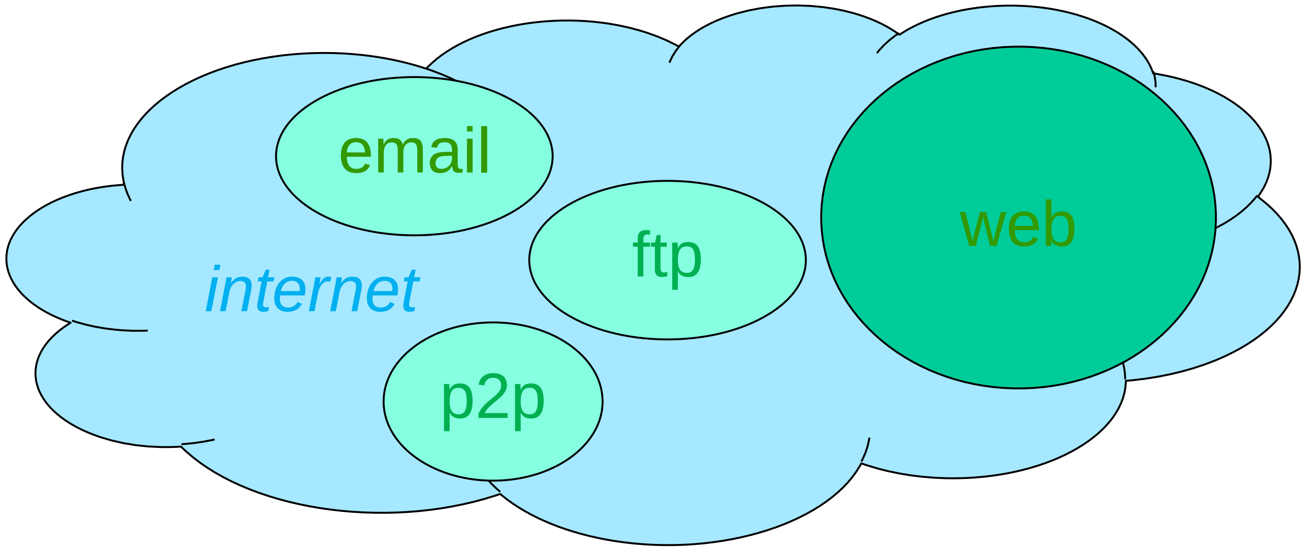
review: **m**odel-**v**iew-**c**ontroller

- **MVC** is a 3-layer pattern



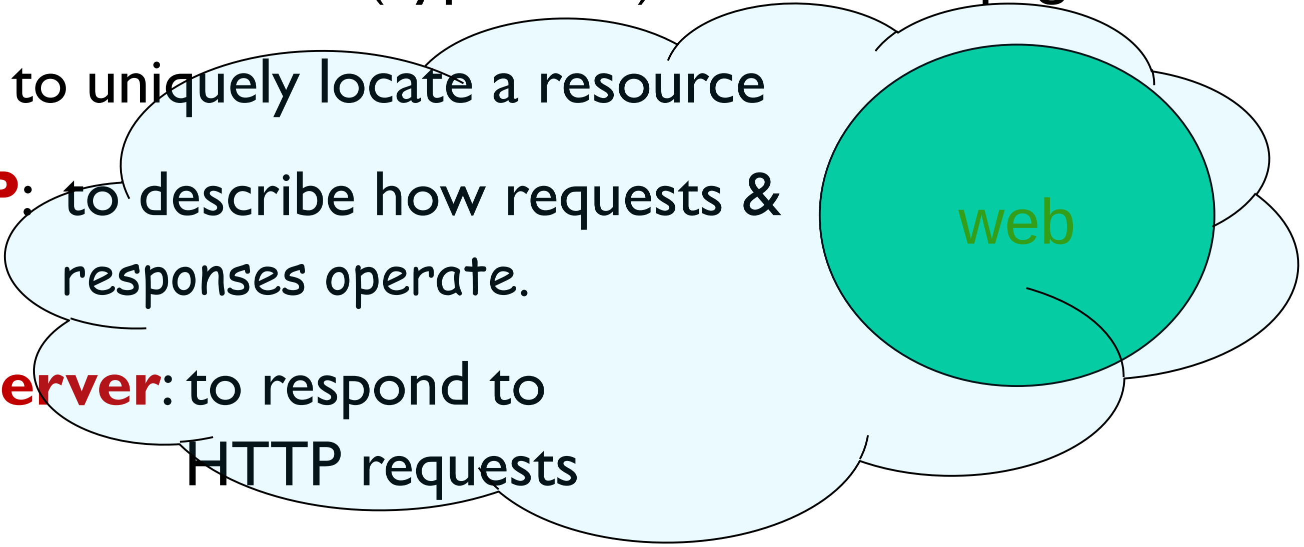
review: internet & services

- is Internet = WWW ?

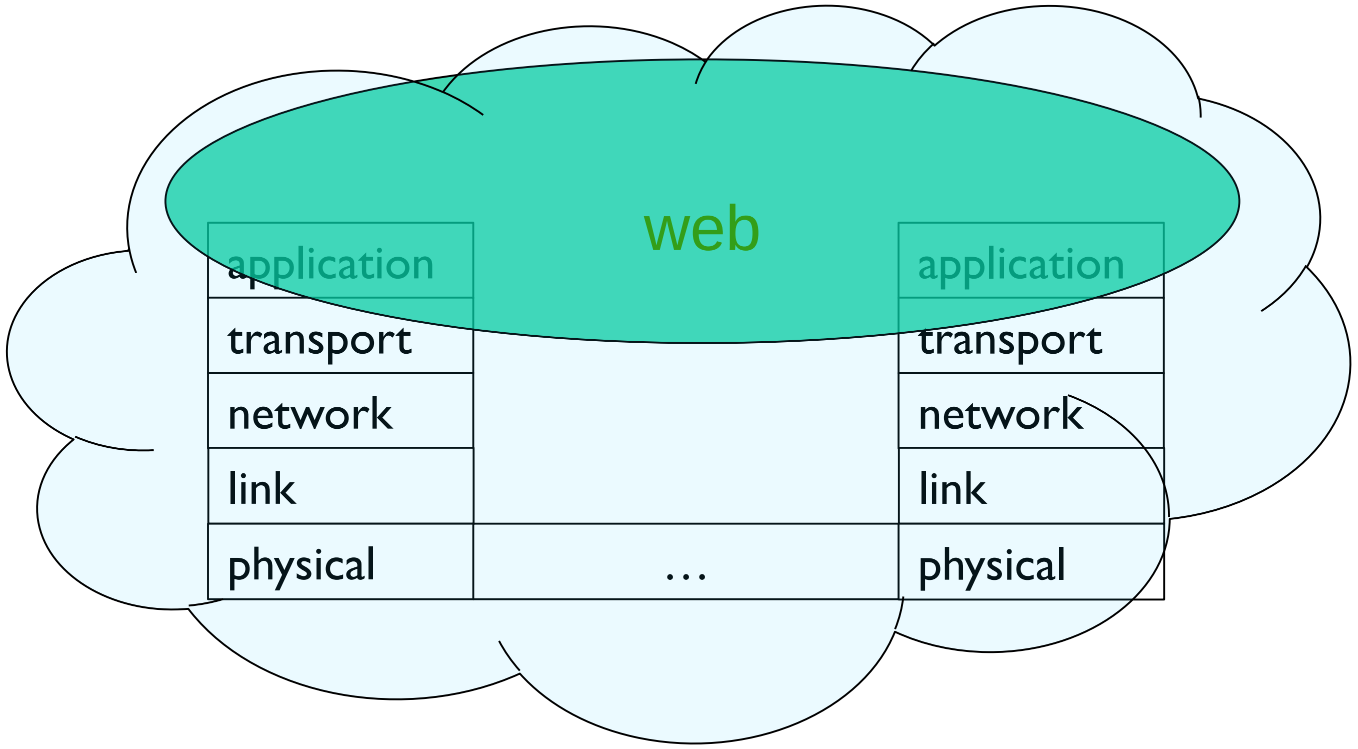


review: www = web

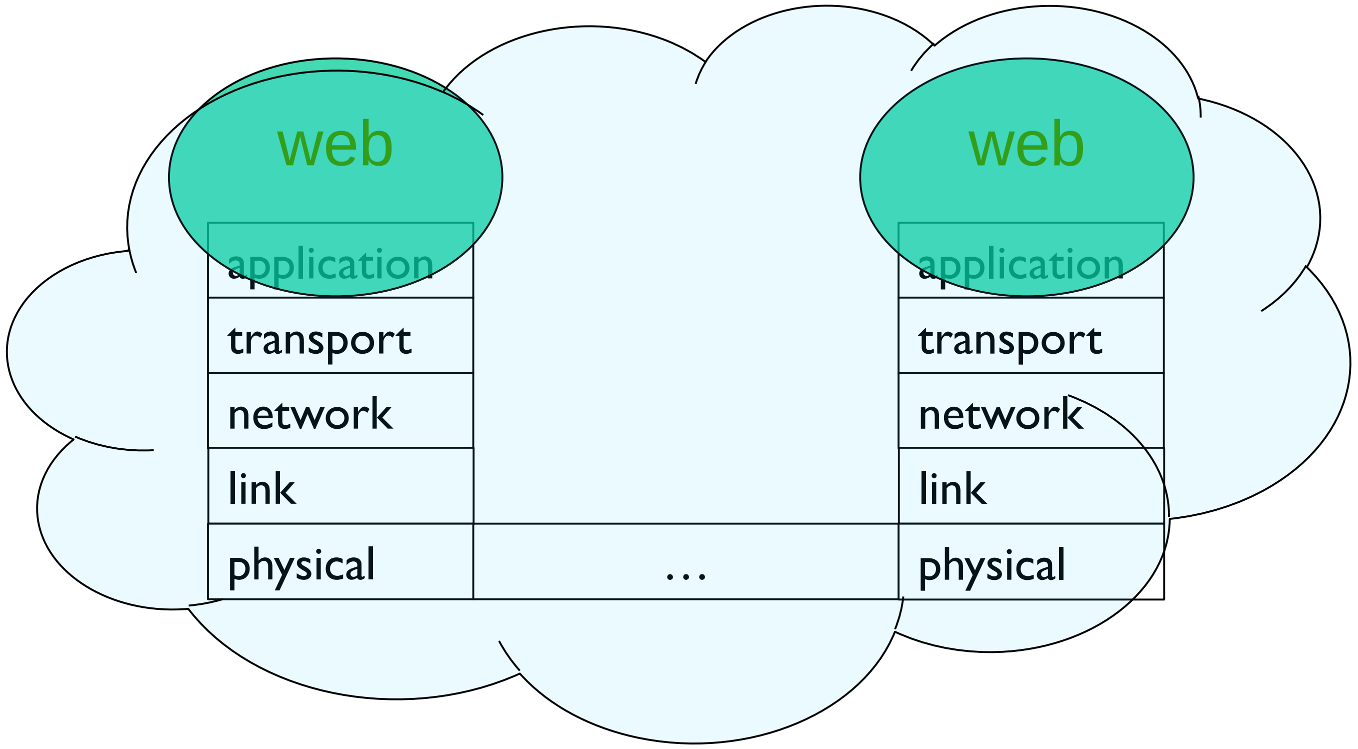
- it's an information space system—based on request & response—with the following features:
  - **HTML**: to describe (hypertext) documents/pages
  - **URL** : to uniquely locate a resource
  - **HTTP**: to describe how requests & responses operate.
  - **web server**: to respond to HTTP requests
  - **web browser**: to make HTTP requests from URLs of servers and render the HTML document received



review: internet layers



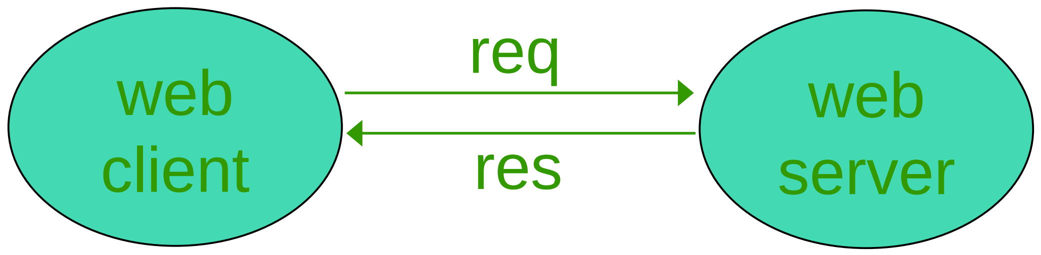
client-server architecture



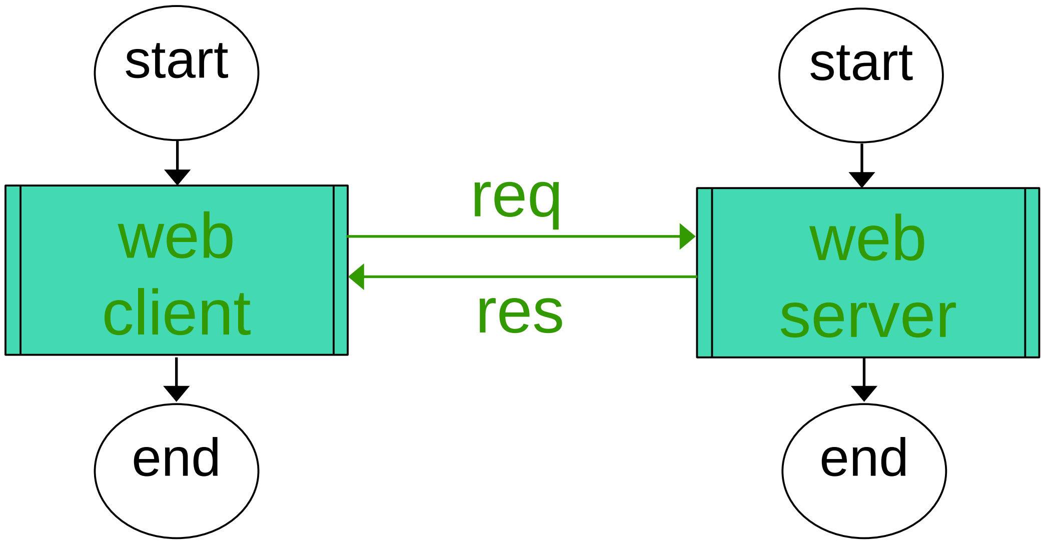
example: codebreaker

web client
web server

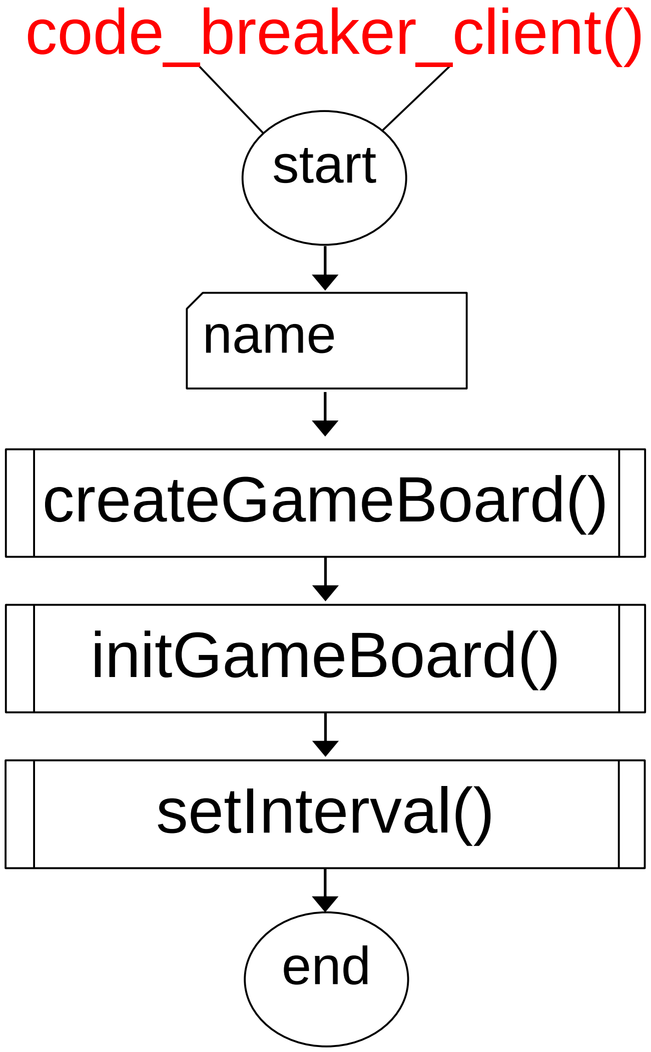
codebreaker design

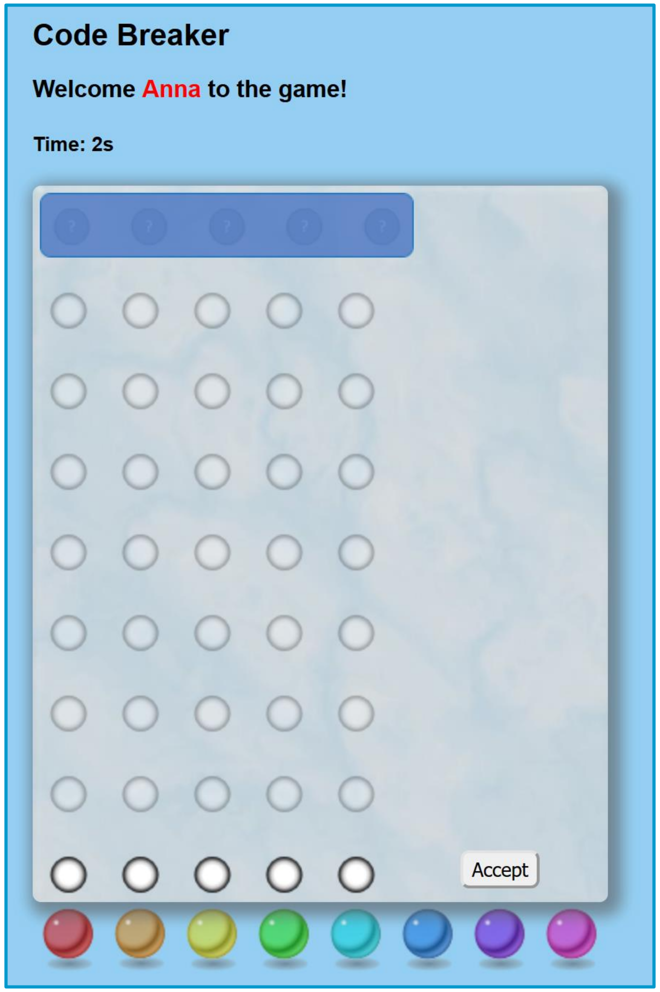


client design

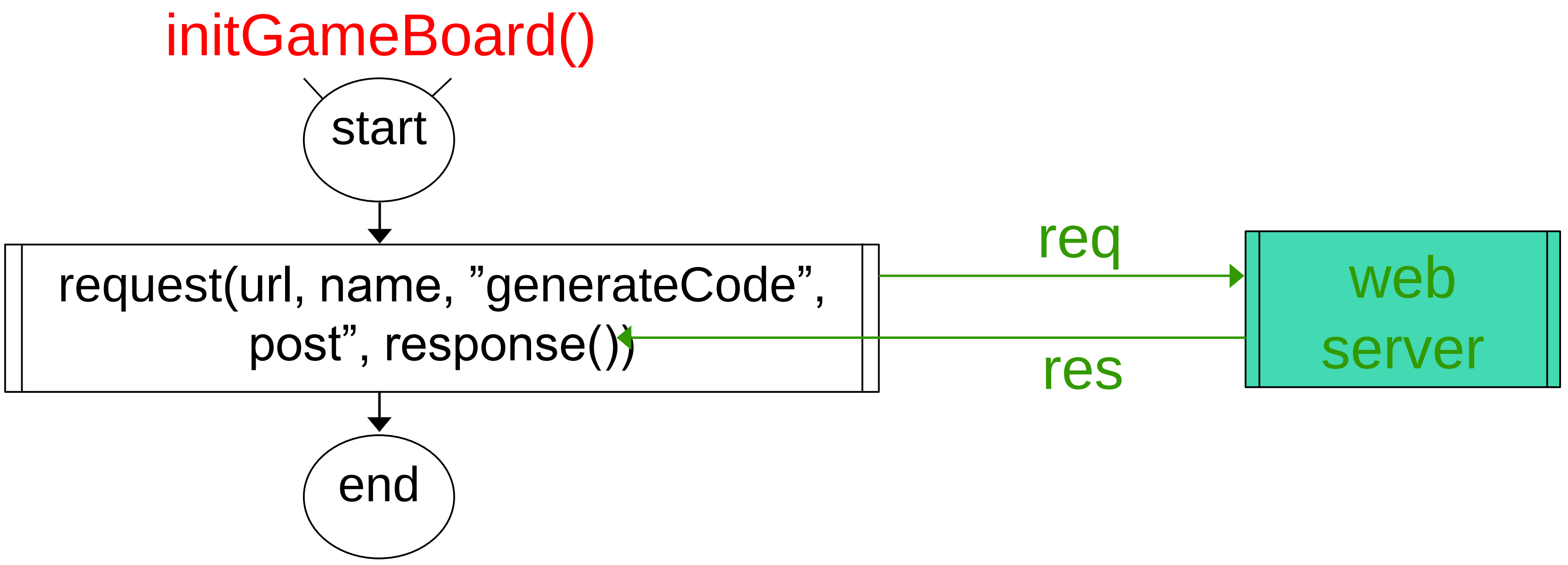


client design

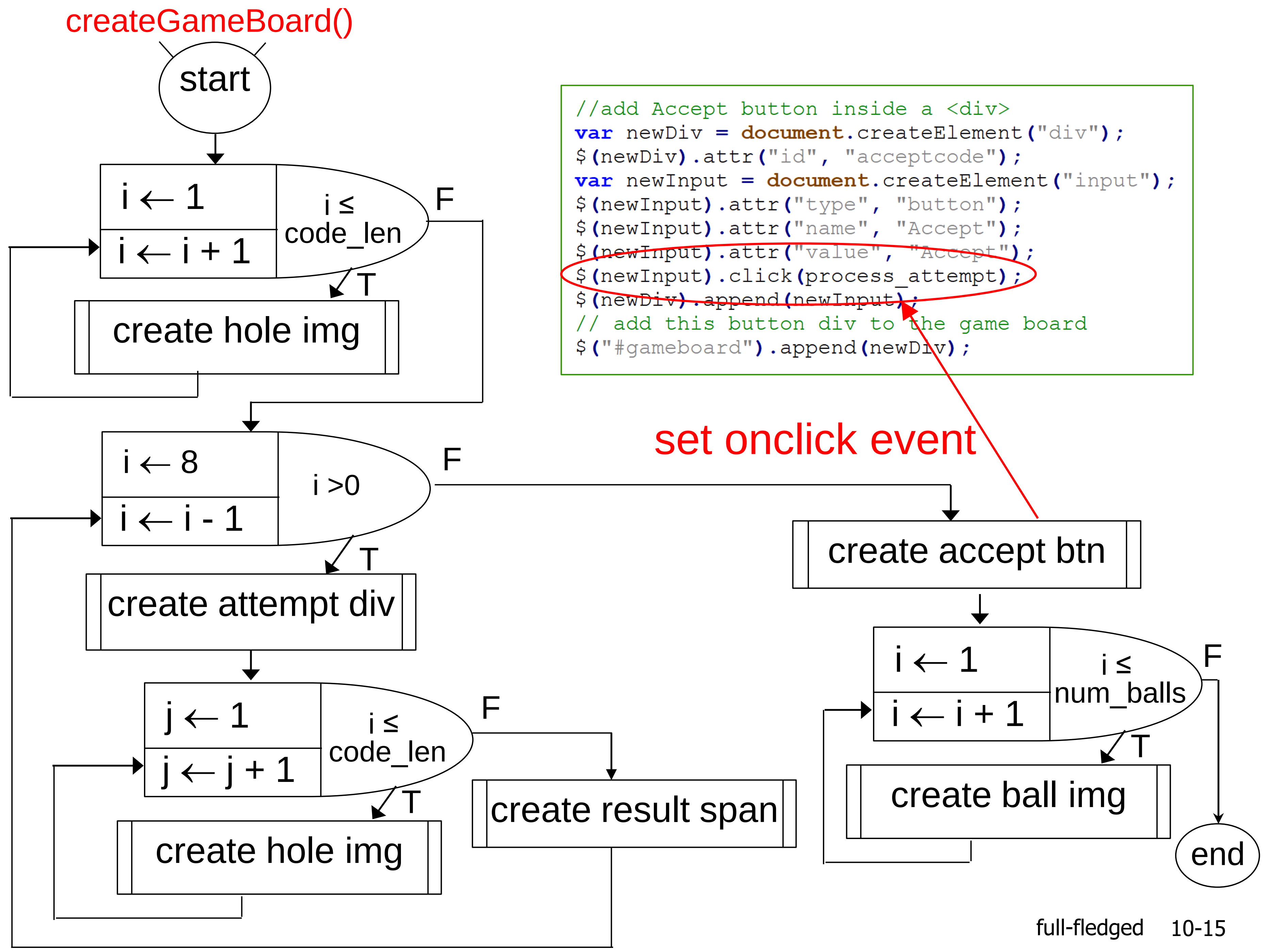




client design

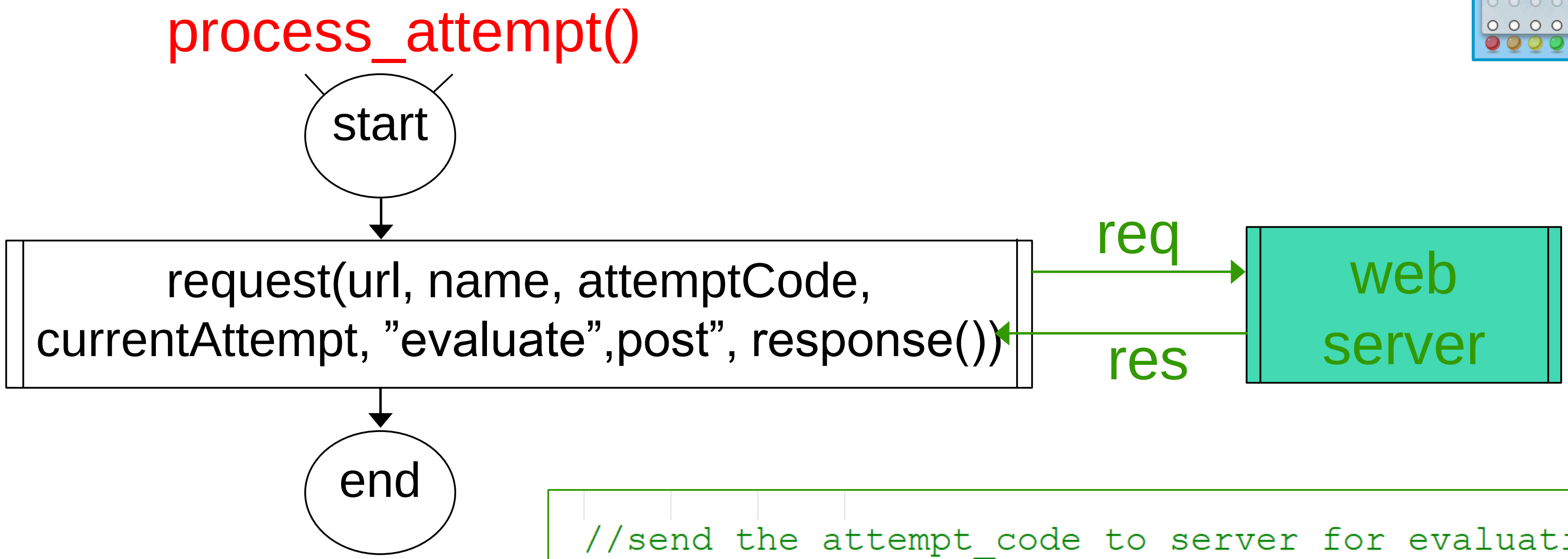


```
//send request to server to start a new game.
$.post(url+'?data='+JSON.stringify({
                        'name':myName,
                        'action':'generateCode'}),
       response);
```



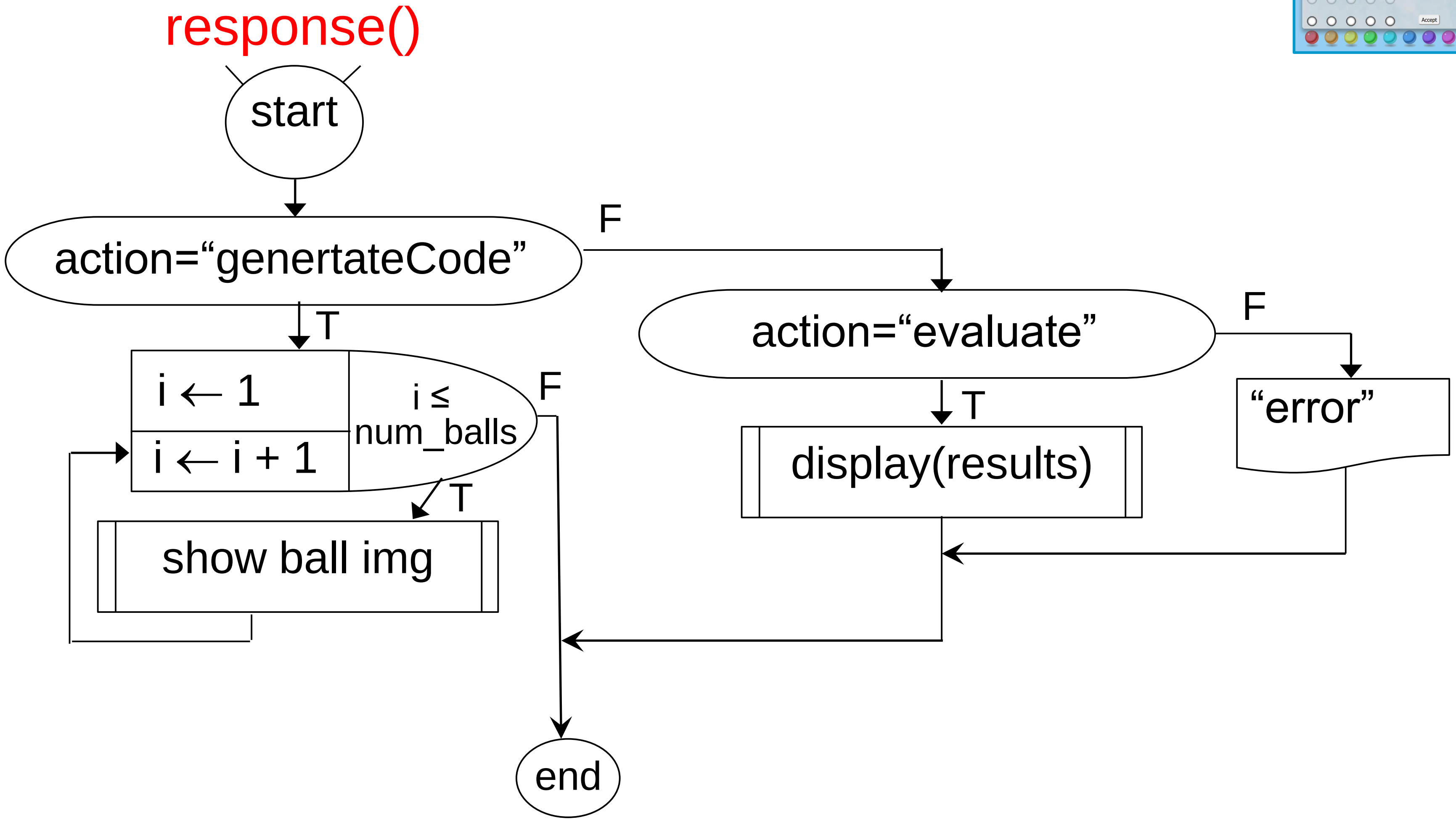
```
//add Accept button inside a <div>
var newDiv = document.createElement("div");
$(newDiv).attr("id", "acceptcode");
var newInput = document.createElement("input");
$(newInput).attr("type", "button");
$(newInput).attr("name", "Accept");
$(newInput).attr("value", "Accept");
$(newInput).click(process_attempt);
$(newDiv).append(newInput);
// add this button div to the game board
$("#gameboard").append(newDiv);
```

client design


process_attempt()
start
request(url, name, attemptCode, currentAttempt, "evaluate",post", response())
req
res
web server
end

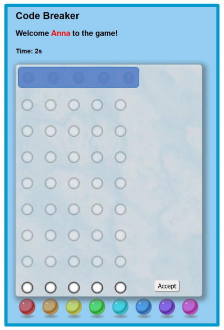
```
//send the attempt_code to server for evaluation
$.post(
    url+'?data='+JSON.stringify({
    'name':myName,
    'action':'evaluate',
    'attempt_code':attempt_code,
    'current_attempt_id':current_attempt_id
    }),
    response
);
```

client design



client design

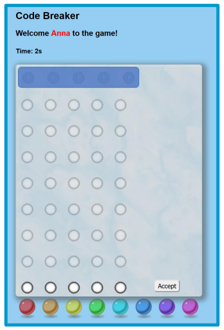
response() … check if action is “**generateCode**”



```
/*
 * Event handler for server's response
 * @param data is the json format string sent from the server
 */
function response(data, status){
    var response = JSON.parse(data);
    console.log(data);
    if (response['action'] == 'generateCode'){
        // acttion: Generate Code
        activateAttempt(1); //activate the first attempt
        peg_selected = 0;   //no peg should be selected
        //reset the visibility of every shadow_balls
        for (var i = 1; i <= NUM_BALLS; i++){
            $("#shadow"+i).css({'opacity' : 1});
        }

        //reset timer
        start = new Date();
```

client design


Code Breaker

Welcome Anna to the game!

Time: 2s

response() … check if action is “**evaluate**”

```
} else if (response['action'] == 'evaluate'){
    // acttion: Evaluate
    // after receiving the server's response,
    // then make the button <div> visible
    $("#acceptcode").css({'visibility' : 'visible'});

    //read data from the json object that send back from the server
    var win = response['win'];
    var num_match = response['num_match'];
    var num_containing = response['num_containing'];
    var num_not_in = response['num_not_in'];
    var code = response['code']

    //display the number of balls that match the code
    displayResult(num_match, "black");
    //display the number of balls in the code
    displayResult(num_containing, "white");
    //display the number of balls not in the code
    displayResult(num_not_in, "empty");
```

client design

response() … check if game has ended

Code Breaker

Welcome Anna to the game!

Time: 2s

```
        if (current_attempt_id < NUM_ATTEMPTS && !win){
            //haven't won yet, game will continue
            //activate the next attempt
            current_attempt_id++;
            activateAttempt(current_attempt_id);
        } else {
            //game ended, display result, hide button
            $("#acceptcode").css({'visibility' : 'hidden'});// hide button <div>
            $("#cover").css({'visibility' : 'hidden'});     // hide code cover to display the code
            displayCode(code);                        // display the code
            win? alert("GG! You win. Click enter to play again.") // won!!!
            : alert("Uh Oh, Click enter to try again!");          // lost!!!
            initGameBoard();
        }
    }
}
```