Server-side programming 2

CS147L Lecture 7
Mike Krieger

Intro

Welcome back!

By the end of today...

- Questions from implementations
- A note on Wizard of Oz
- Debugging PHP & Javascript
- More SQLite examples
- Full app: Where my friends at?

Administrative Stuff

- One more lecture next week
- No lecture dead week
- Email me with an honor code statement stating you attended 7/8 of the lectures

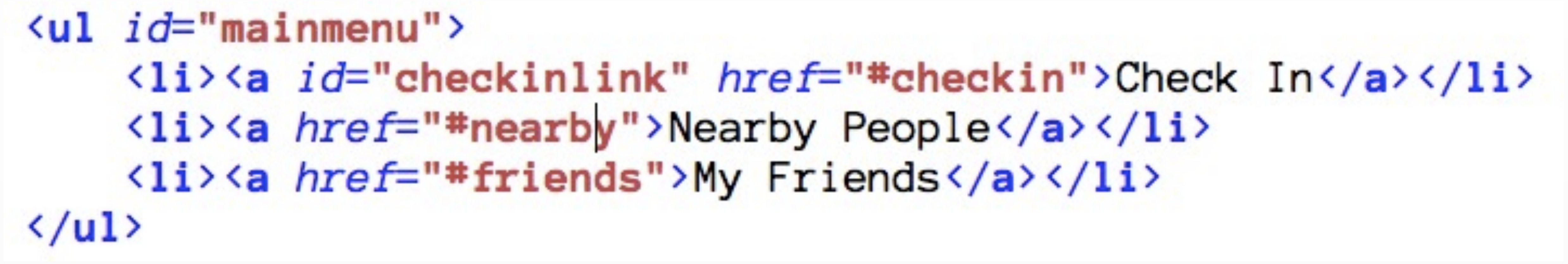
Question from Google Group

Action on link before changing page?

- http://groups.google.com/group/cs147-tech-2009/browse_thread/thread/7b26b35f41041a28

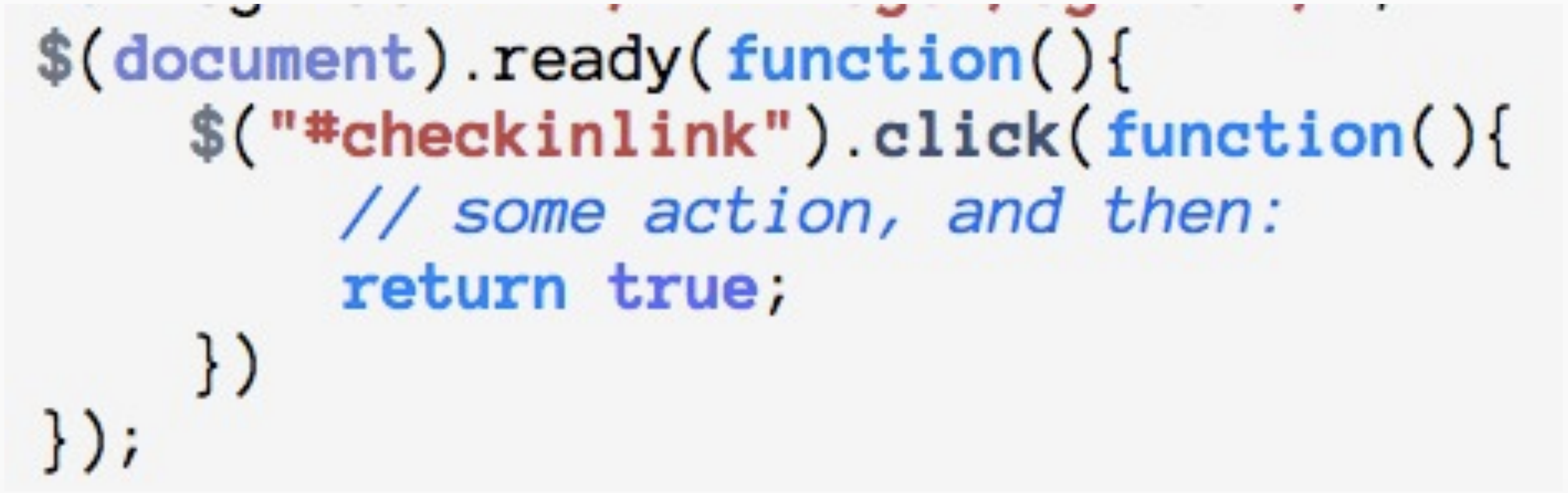
Solution

Give link an id:



```
<ul id="mainmenu">
    <li><a id="checkinlink" href="#checkin">Check In</a></li>
    <li><a href="#nearby">Nearby People</a></li>
    <li><a href="#friends">My Friends</a></li>
</ul>
```

Respond & return true:



```
$(document).ready(function(){
    $("#checkinlink").click(function(){
        // some action, and then:
        return true;
    })
});
```

Any other current Qs?

A note on WoZ

Wizard of Oz

- Have gotten some questions about camera, SMS, phone...

My app needs to take a picture

- *Problem*: iPod Touch has no camera
- *WoZ*: Take a screenshot of camera app from iPhone, bring it up at appropriate moment (or a video of the camera app moving around)
- I put some screenshots & video in the "resources" section

My app needs to make a call

- *Problem*: iPod Touch has no phone app
- *WoZ*: Present fake phone call screen (or build quick one in JS), have users act out call

App needs SMS send/ receive

- *WoZ*: pop up Javascript alert() or prompt () with SMS message, have your app respond

Debugging Webapps

Quick Note

- WebKit nightly builds have terrific debugging tools
    - http://webkit.org
- Windows, Mac, & Linux

Key Questions

- Is the right data living on the server?
- Is the right data making it back from the server?
- Is the Javascript parsing the data correctly?

Is the right data living on the server?

- Run same SELECT command using command-line tool

dumpdb.php

```
In this week's folder, will go through all your entries in all
your tables and print them out.
```

Running PHP from commandline

```
php yourfilename.php
```

No browser required!

Is it making it back?

- Just browser will work
- On Mac, can also use HTTP Analyzer
- Use Inspector's Request view

Demos

```
Just browser (friends-1.php)
HTTP Client
Safari Inspector
```

Debugging Javascript

General approach

- Set *breakpoints* in Safari
- Use *console.log* when breakpoints are impractical / too much work

Demos

`debug-1 to debug-3.html`

FriendFinder

Demo app features

- Locate user (we know how to do this)
- Update a central DB of locations
- Show:
    - People around me
    - Friend list

Approach

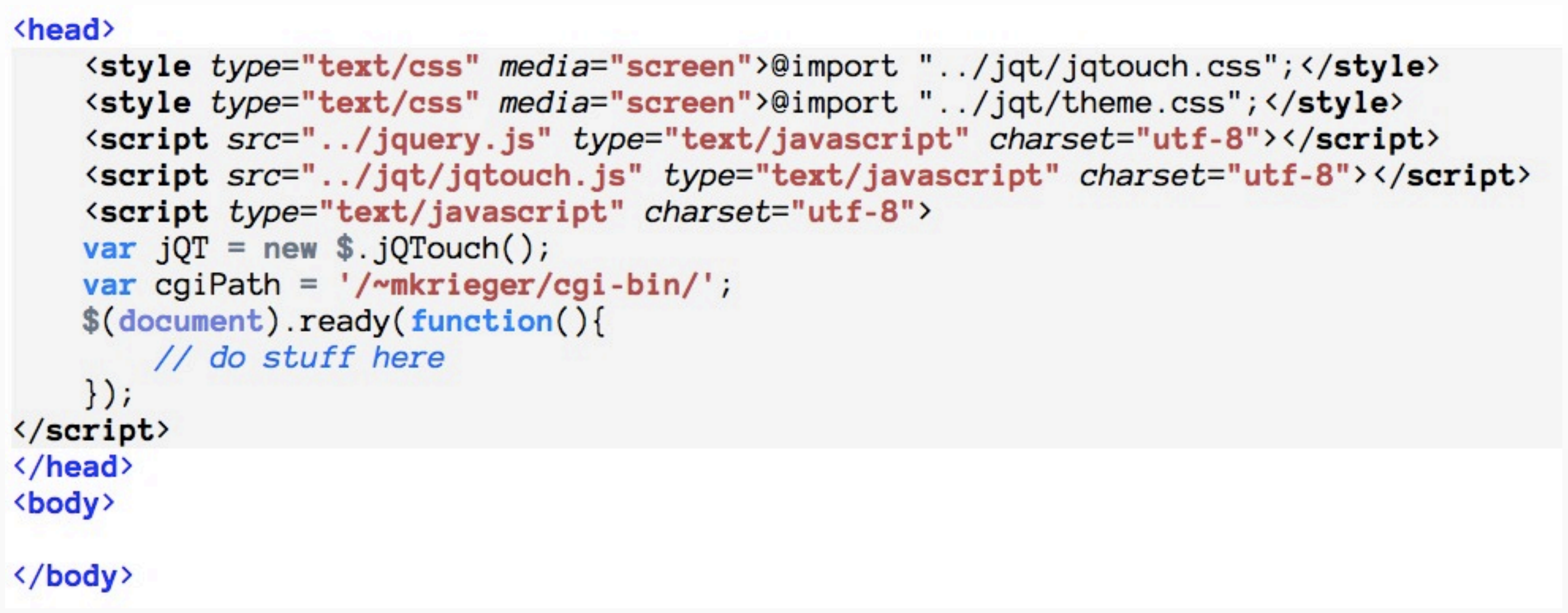
- Write JS and PHP separately, join near the end (you can do this on your team too)
- Use browser/HTTP Client to debug PHP calls
- Use Inspector to debug & develop JS & use fake data in meantime

Getting started

friends-0

- jQTouch template

friends-0.html

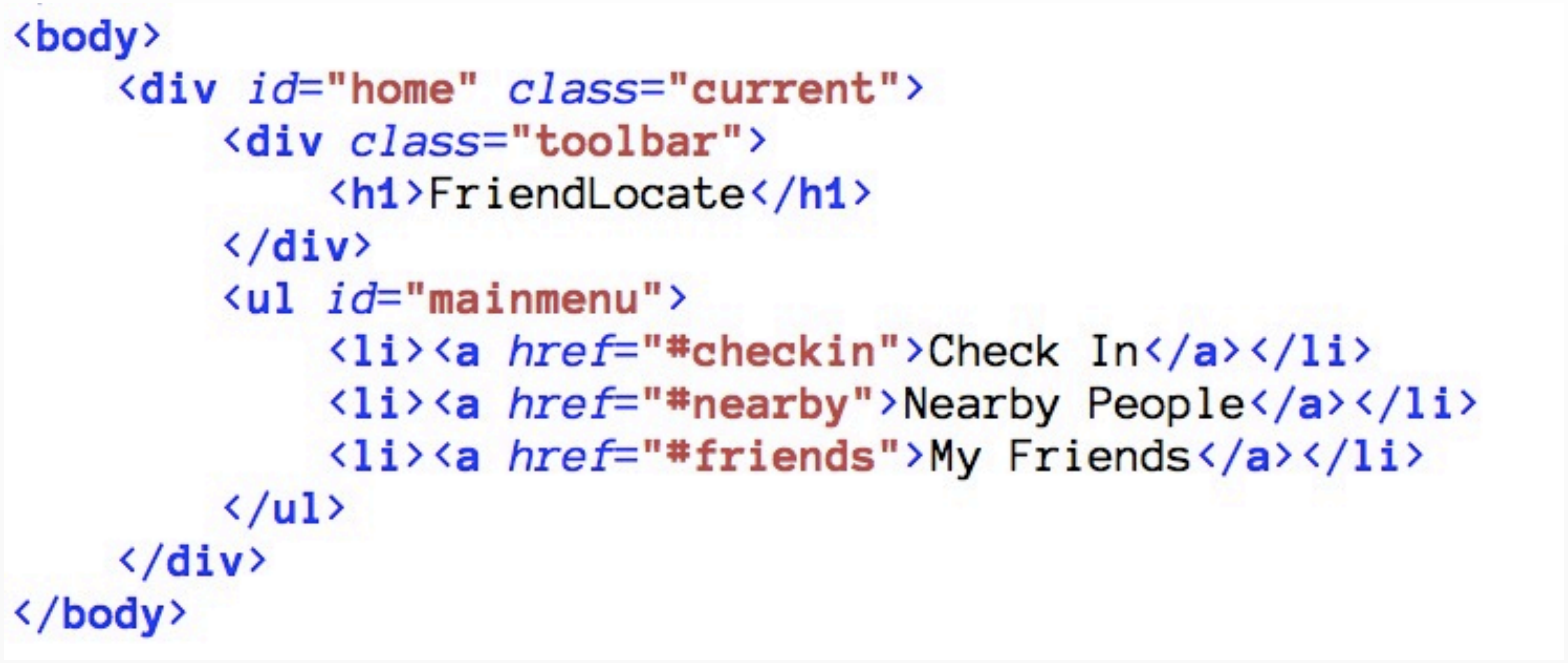


```
<head>
    <style type="text/css" media="screen">@import "../jqt/jqtouch.css";</style>
    <style type="text/css" media="screen">@import "../jqt/theme.css";</style>
    <script src="../jquery.js" type="text/javascript" charset="utf-8"></script>
    <script src="../jqt/jqtouch.js" type="text/javascript" charset="utf-8"></script>
    <script type="text/javascript" charset="utf-8">
    var jQT = new $.jQTouch();
    var cgiPath = '/~mkrieger/cgi-bin/';
    $(document).ready(function(){
        // do stuff here
    });
</script>
</head>
<body>

</body>
```

friends-1.html

- Skeleton menu

friends-1.html

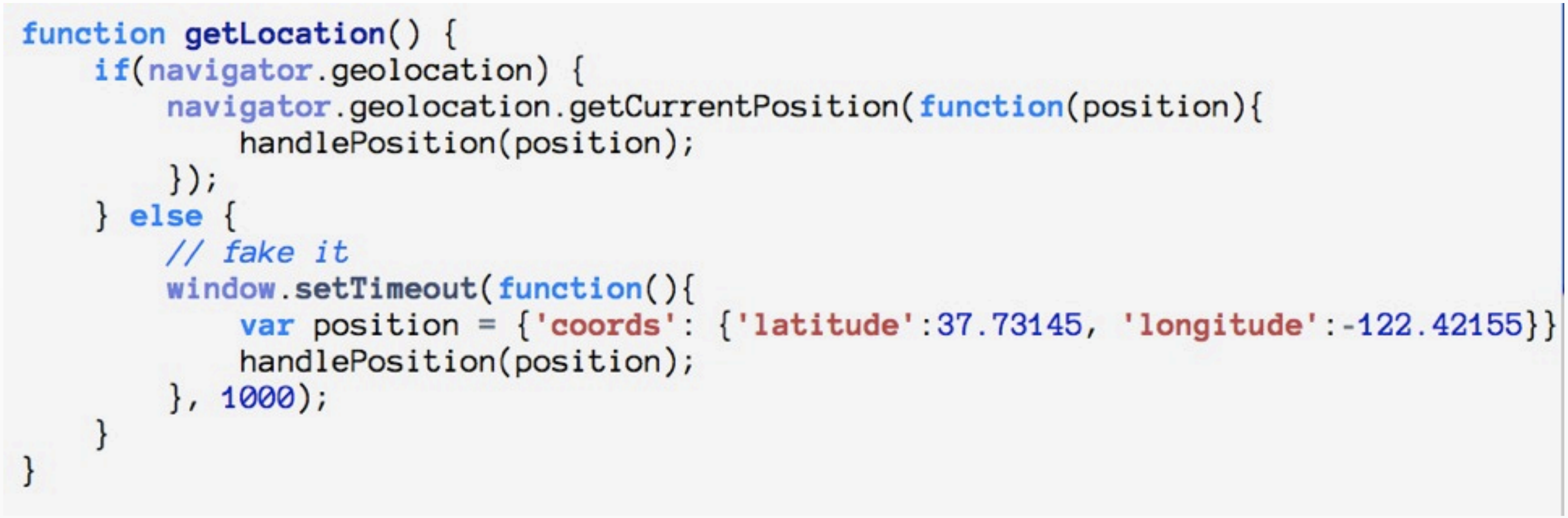


```html
<body>
    <div id="home" class="current">
        <div class="toolbar">
            <h1>FriendLocate</h1>
        </div>
        <ul id="mainmenu">
            <li><a href="#checkin">Check In</a></li>
            <li><a href="#nearby">Nearby People</a></li>
            <li><a href="#friends">My Friends</a></li>
        </ul>
    </div>
</body>
```

friends-2.html

- Get location
- Get a user's status

Now-familiar location



```
function getLocation() {
    if(navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(function(position){
            handlePosition(position);
        });
    } else {
        // fake it
        window.setTimeout(function(){
            var position = {'coords': {'latitude':37.73145, 'longitude':-122.42155}}
            handlePosition(position);
        }, 1000);
    }
}
```

prompt()

- Takes a string, returns whatever was entered into the prompt window

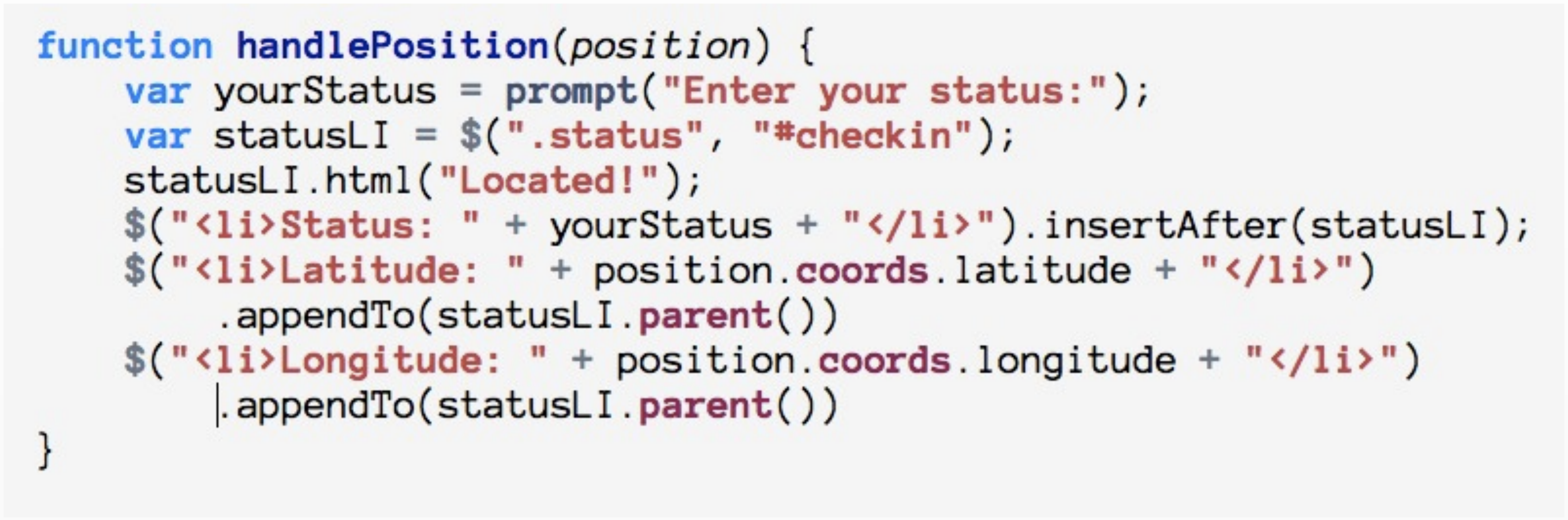
Callback

```
function handlePosition(position) {
    var yourStatus = prompt("Enter your status:");
    var statusLI = $(".status", "#checkin");
    statusLI.html("Located!");
    $("<li>Status: " + yourStatus + "</li>").insertAfter(statusLI);
}
```

friends-3.html

- Provide confirmation that the status was sent to server

Feedback



```
function handlePosition(position) {
    var yourStatus = prompt("Enter your status:");
    var statusLI = $(".status", "#checkin");
    statusLI.html("Located!");
    $("<li>Status: " + yourStatus + "</li>").insertAfter(statusLI);
    $("<li>Latitude: " + position.coords.latitude + "</li>")
        .appendTo(statusLI.parent())
    $("<li>Longitude: " + position.coords.longitude + "</li>")
        .appendTo(statusLI.parent())
}
```

friends-4.html

- Get a username and save it for this particular user using window.localStorage

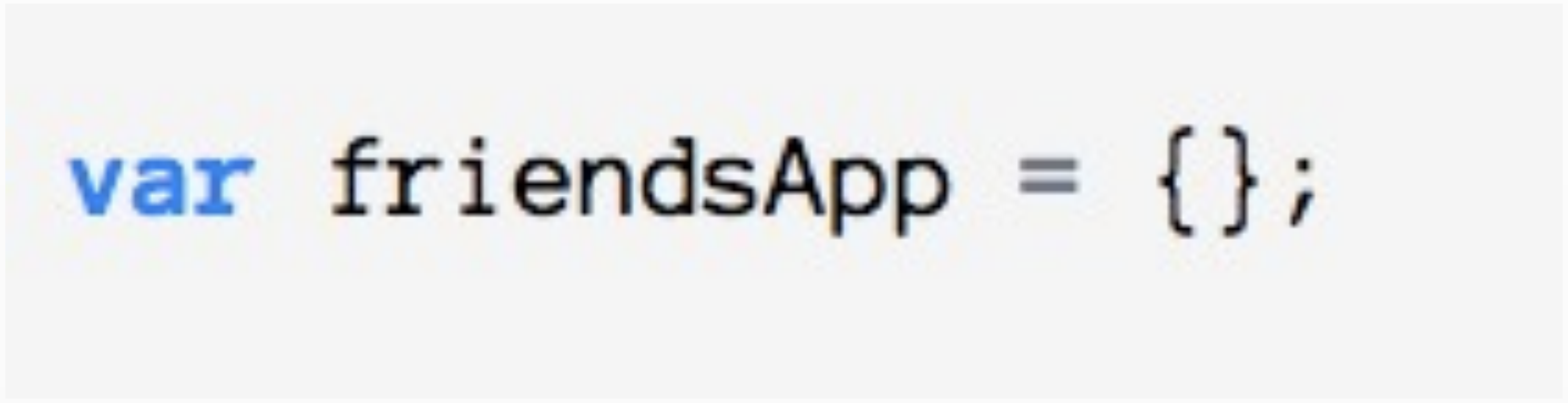
localStorage

- HTML5 specification
- Provides key/value store
- Easier & more robust than using cookies

Two functions

```
window.localStorage.getItem("itemName");

window.localStorage.setItem("itemName", "value");

// example:

if (!window.localStorage.getItem("city")) {
   // prompt user for city
   var city = prompt("What city are you in?");
   window.localStorage.setItem("city", city);
}
```

global object

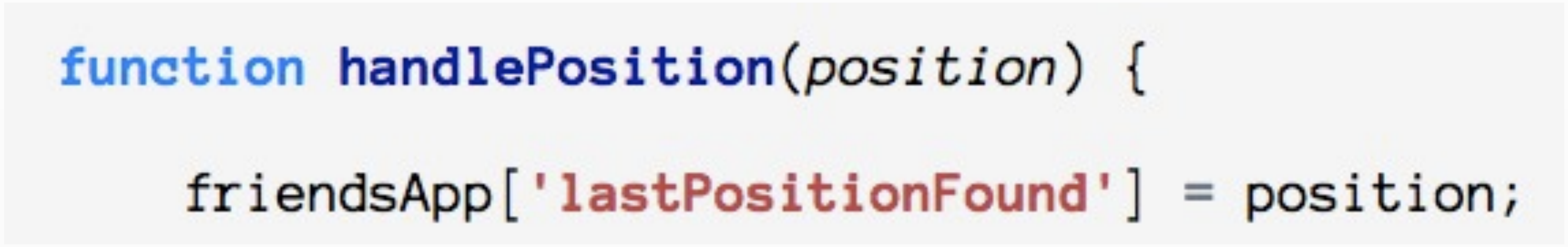


```
var friendsApp = {};
var username = window.localStorage.getItem("username");

if (!username) {
    username = window.prompt("Your name?");
    window.localStorage.setItem("username", username);
}

friendsApp['username'] = username;
```

friends-5.php

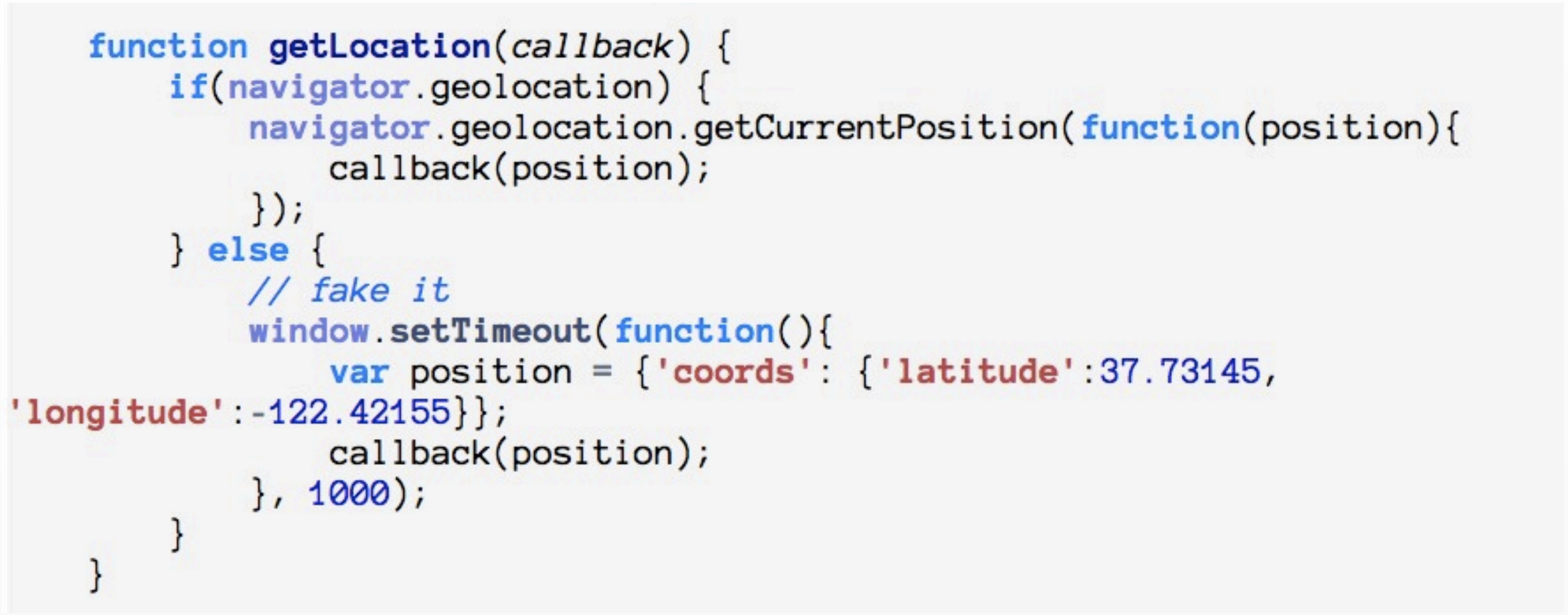
- Cache the found location globally so we don't have to fetch it every time we switch pages
- Have getLocation handle different callbacks

Save Position



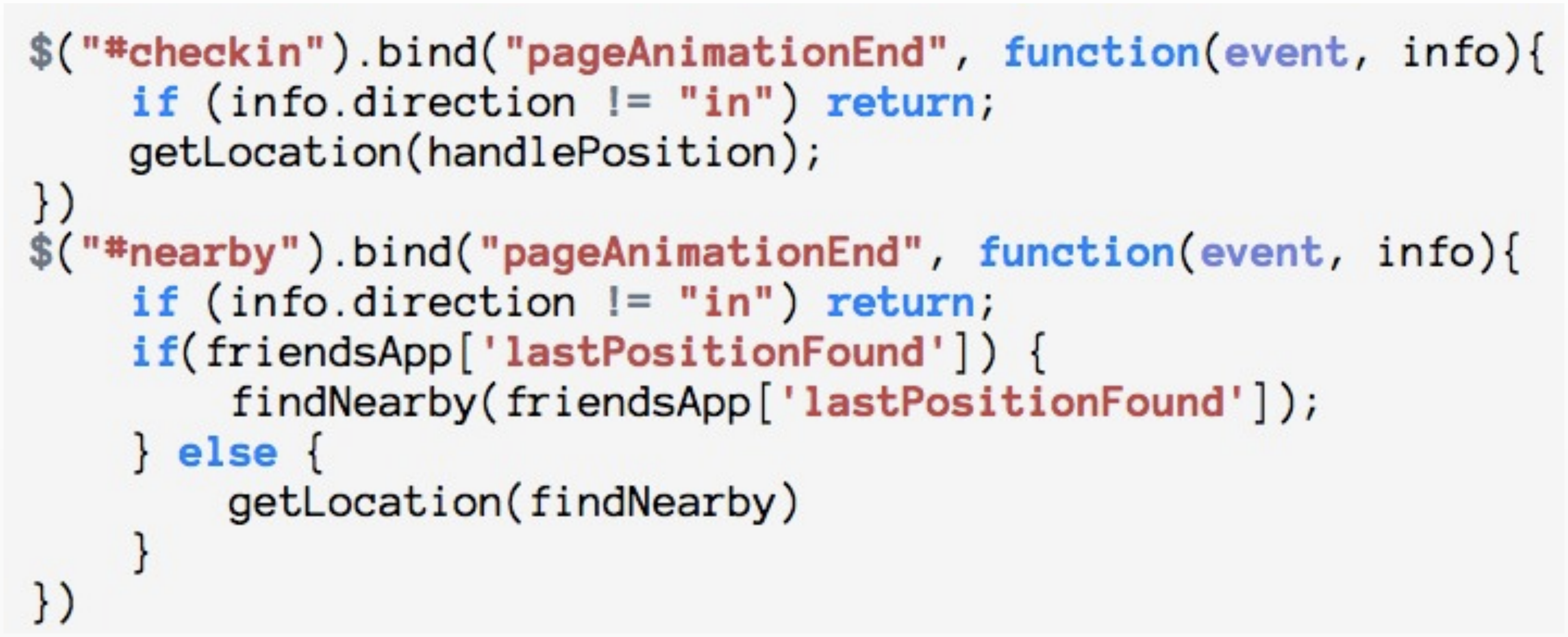
```
function handlePosition(position) {

    friendsApp['lastPositionFound'] = position;
```

Callbacks



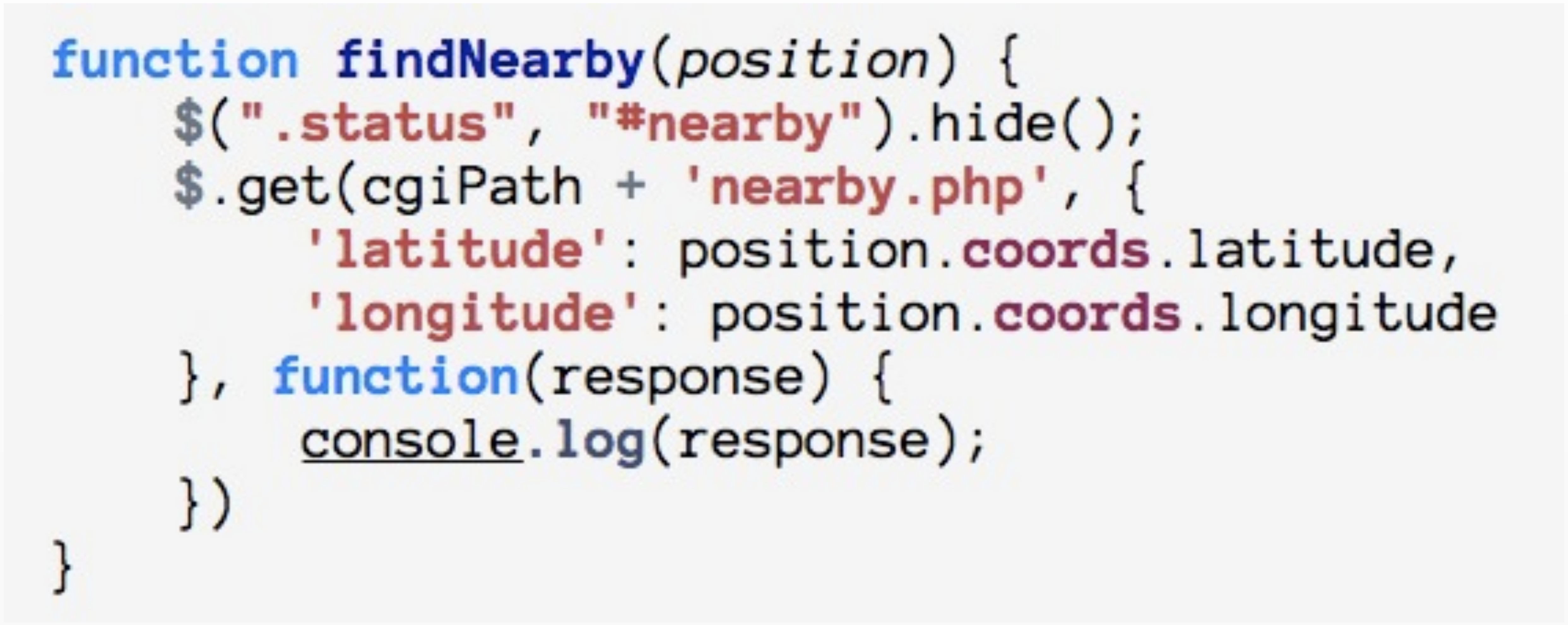
```
    function getLocation(callback) {
        if(navigator.geolocation) {
            navigator.geolocation.getCurrentPosition(function(position){
                callback(position);
            });
        } else {
            // fake it
            window.setTimeout(function(){
                var position = {'coords': {'latitude':37.73145,
'longitude':-122.42155}};
                callback(position);
            }, 1000);
        }
    }
```

Flexibility


```
$("#checkin").bind("pageAnimationEnd", function(event, info){
    if (info.direction != "in") return;
    getLocation(handlePosition);
})
$("#nearby").bind("pageAnimationEnd", function(event, info){
    if (info.direction != "in") return;
    if(friendsApp['lastPositionFound']) {
        findNearby(friendsApp['lastPositionFound']);
    } else {
        getLocation(findNearby)
    }
})
```

friends-6

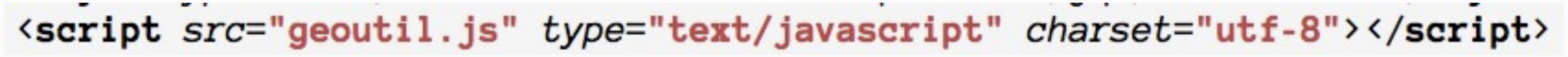
- Beginning of findNearby function



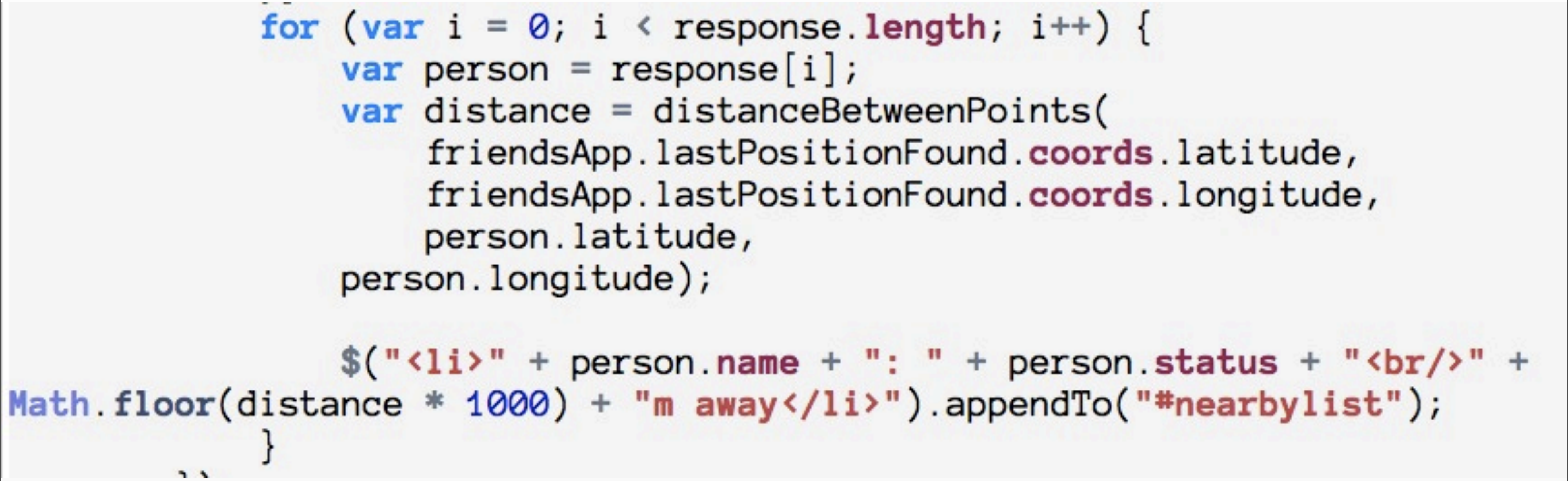
```
function findNearby(position) {
    $(".status", "#nearby").hide();
    $.get(cgiPath + 'nearby.php', {
        'latitude': position.coords.latitude,
        'longitude': position.coords.longitude
    }, function(response) {
         console.log(response);
    })
}
```

friends-7.php

- Import geoutil from last week's code
- Adding placeholder JSON object for response



```
<script src="geoutil.js" type="text/javascript" charset="utf-8"></script>
```



```
            for (var i = 0; i < response.length; i++) {
                var person = response[i];
                var distance = distanceBetweenPoints(
                    friendsApp.lastPositionFound.coords.latitude,
                    friendsApp.lastPositionFound.coords.longitude,
                    person.latitude,
                person.longitude);

                $("<li>" + person.name + ": " + person.status + "<br/>" +
Math.floor(distance * 1000) + "m away</li>").appendTo("#nearbylist");
            }
```

friends-8.php

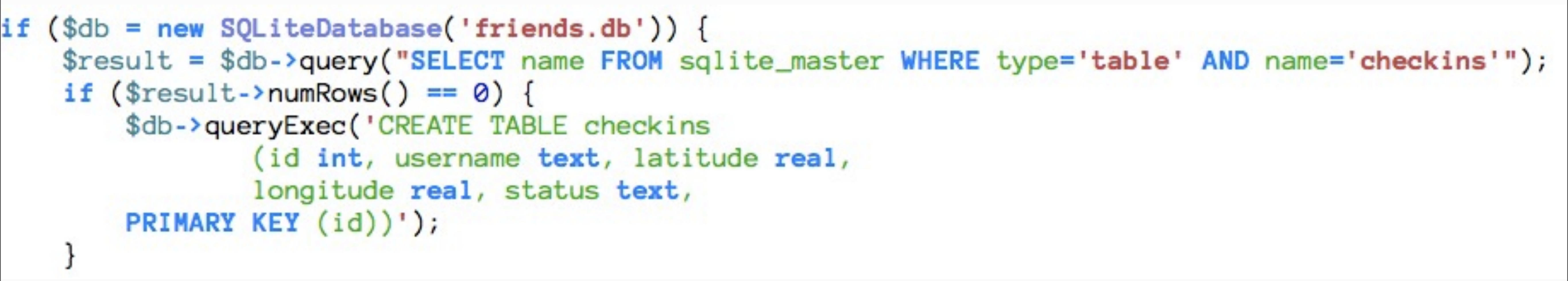
- getFriends() function
-

Server-side

Preparing the DB

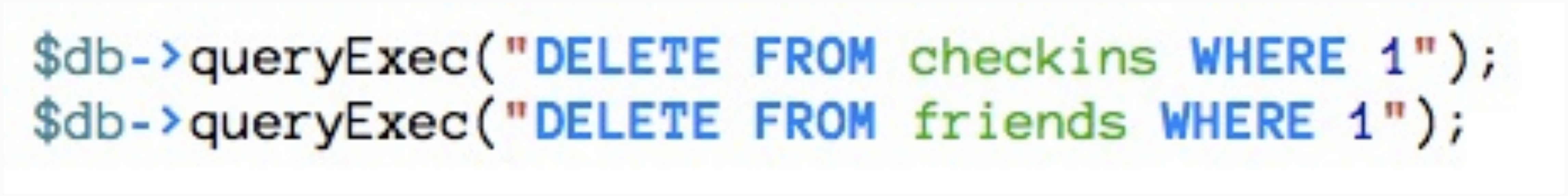
- preparedb.php

Make the tables



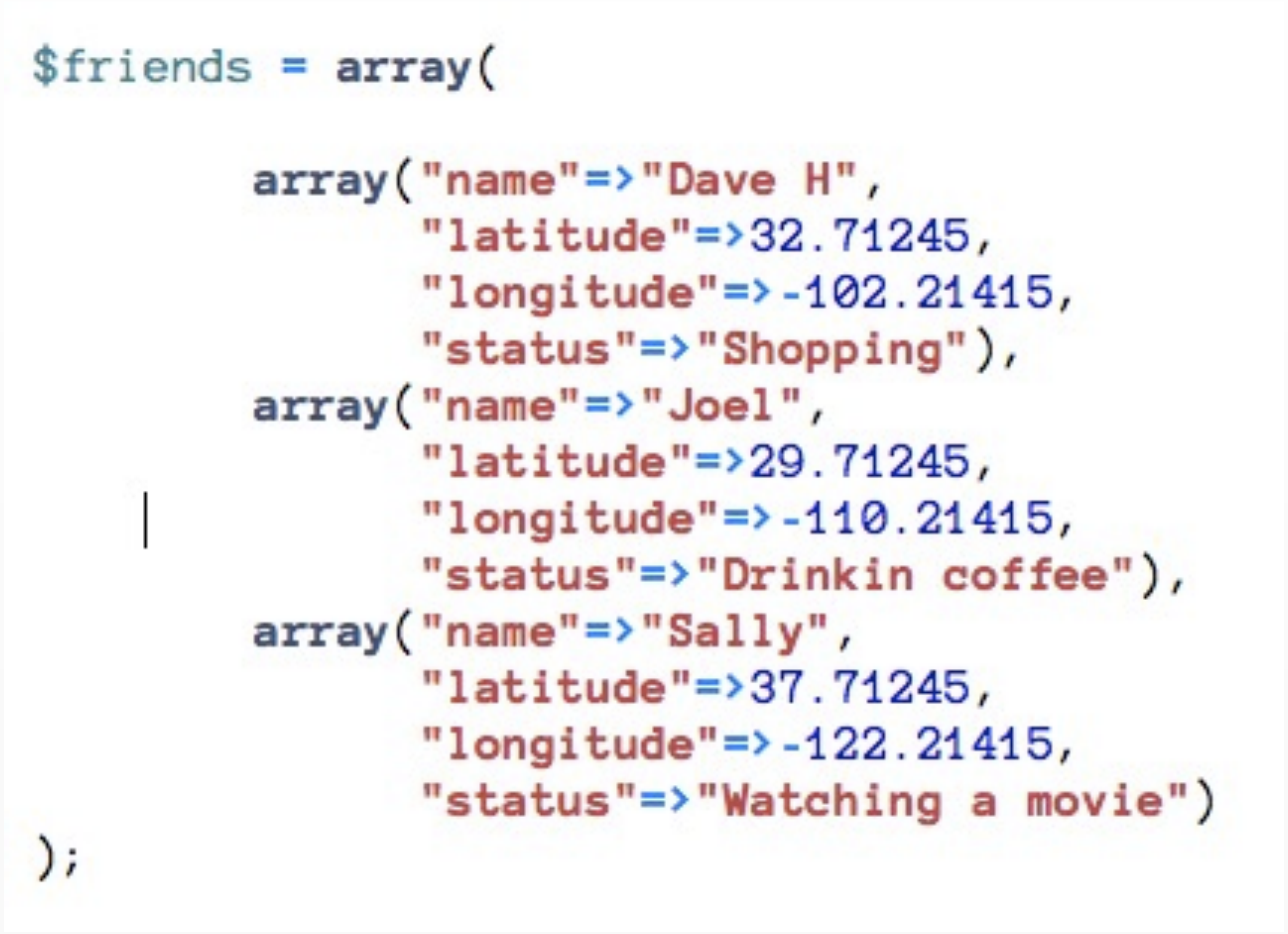
```
if ($db = new SQLiteDatabase('friends.db')) {
    $result = $db->query("SELECT name FROM sqlite_master WHERE type='table' AND name='checkins'");
    if ($result->numRows() == 0) {
        $db->queryExec('CREATE TABLE checkins
                (id int, username text, latitude real,
                longitude real, status text,
        PRIMARY KEY (id))');
    }
```

Clear the tables



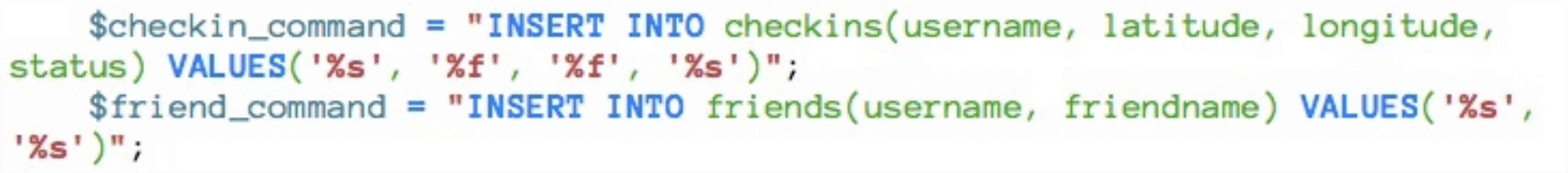
```
$db->queryExec("DELETE FROM checkins WHERE 1");
$db->queryExec("DELETE FROM friends WHERE 1");
```

Fixture data



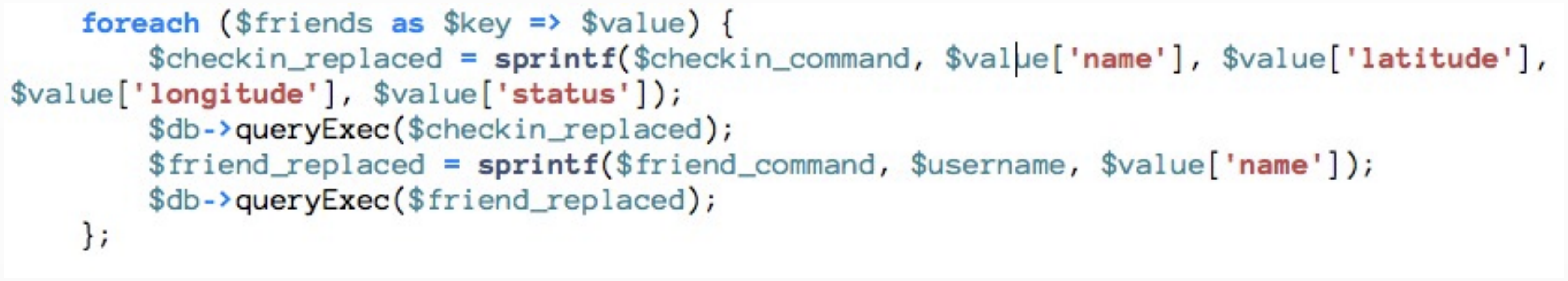
```
$friends = array(

        array("name"=>"Dave H",
              "latitude"=>32.71245,
              "longitude"=>-102.21415,
              "status"=>"Shopping"),
        array("name"=>"Joel",
              "latitude"=>29.71245,
              "longitude"=>-110.21415,
              "status"=>"Drinkin coffee"),
        array("name"=>"Sally",
              "latitude"=>37.71245,
              "longitude"=>-122.21415,
              "status"=>"Watching a movie")
);
```

DB Commands



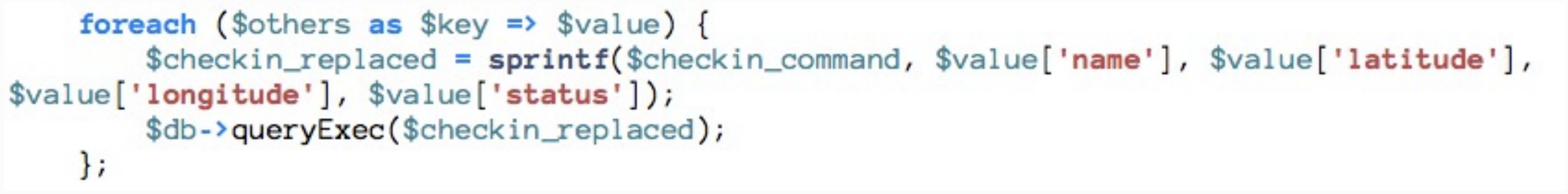
```
    $checkin_command = "INSERT INTO checkins(username, latitude, longitude,
status) VALUES('%s', '%f', '%f', '%s')";
    $friend_command = "INSERT INTO friends(username, friendname) VALUES('%s',
'%s')";
```

Iterate over friends



```
    foreach ($friends as $key => $value) {
        $checkin_replaced = sprintf($checkin_command, $value['name'], $value['latitude'],
$value['longitude'], $value['status']);
        $db->queryExec($checkin_replaced);
        $friend_replaced = sprintf($friend_command, $username, $value['name']);
        $db->queryExec($friend_replaced);
    };
```

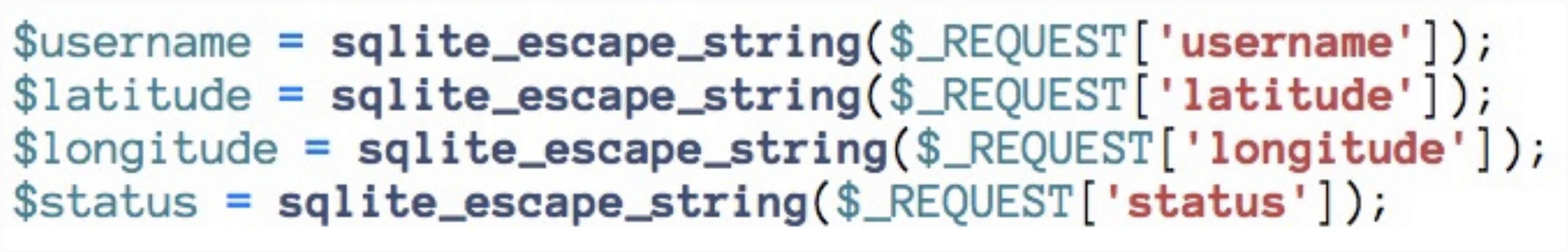
Iterate over others



```
    foreach ($others as $key => $value) {
        $checkin_replaced = sprintf($checkin_command, $value['name'], $value['latitude'],
$value['longitude'], $value['status']);
        $db->queryExec($checkin_replaced);
    };
```

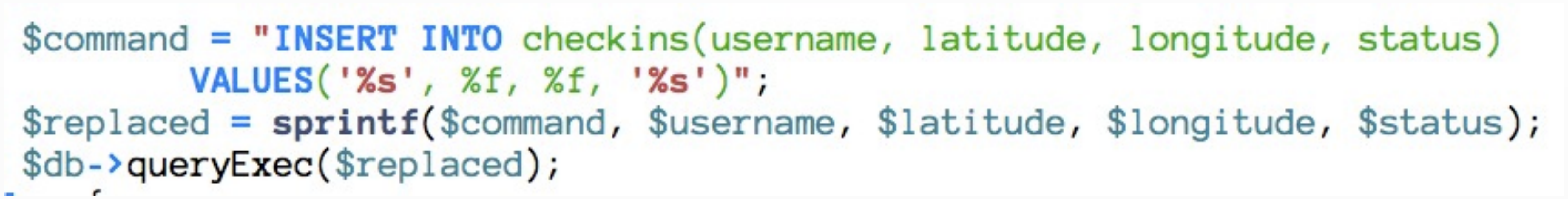
checkin.php

```
// grab data
```



```php
$username = sqlite_escape_string($_REQUEST['username']);
$latitude = sqlite_escape_string($_REQUEST['latitude']);
$longitude = sqlite_escape_string($_REQUEST['longitude']);
$status = sqlite_escape_string($_REQUEST['status']);
```

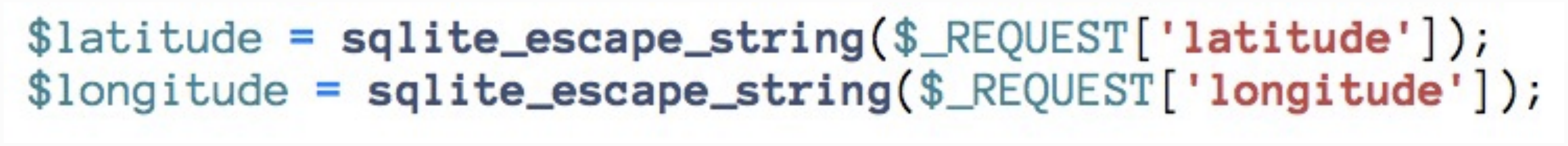
Insert row



```
$command = "INSERT INTO checkins(username, latitude, longitude, status)
        VALUES('%s', %f, %f, '%s')";
$replaced = sprintf($command, $username, $latitude, $longitude, $status);
$db->queryExec($replaced);
```

nearby.php

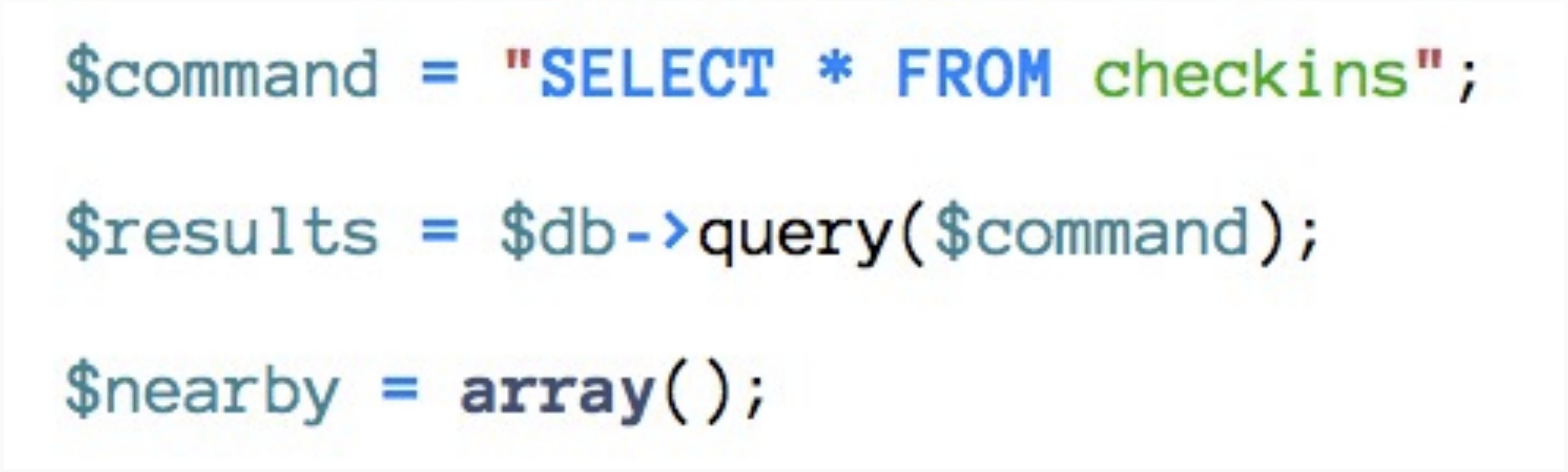
```
// grab params
```



```php
$latitude = sqlite_escape_string($_REQUEST['latitude']);
$longitude = sqlite_escape_string($_REQUEST['longitude']);
```

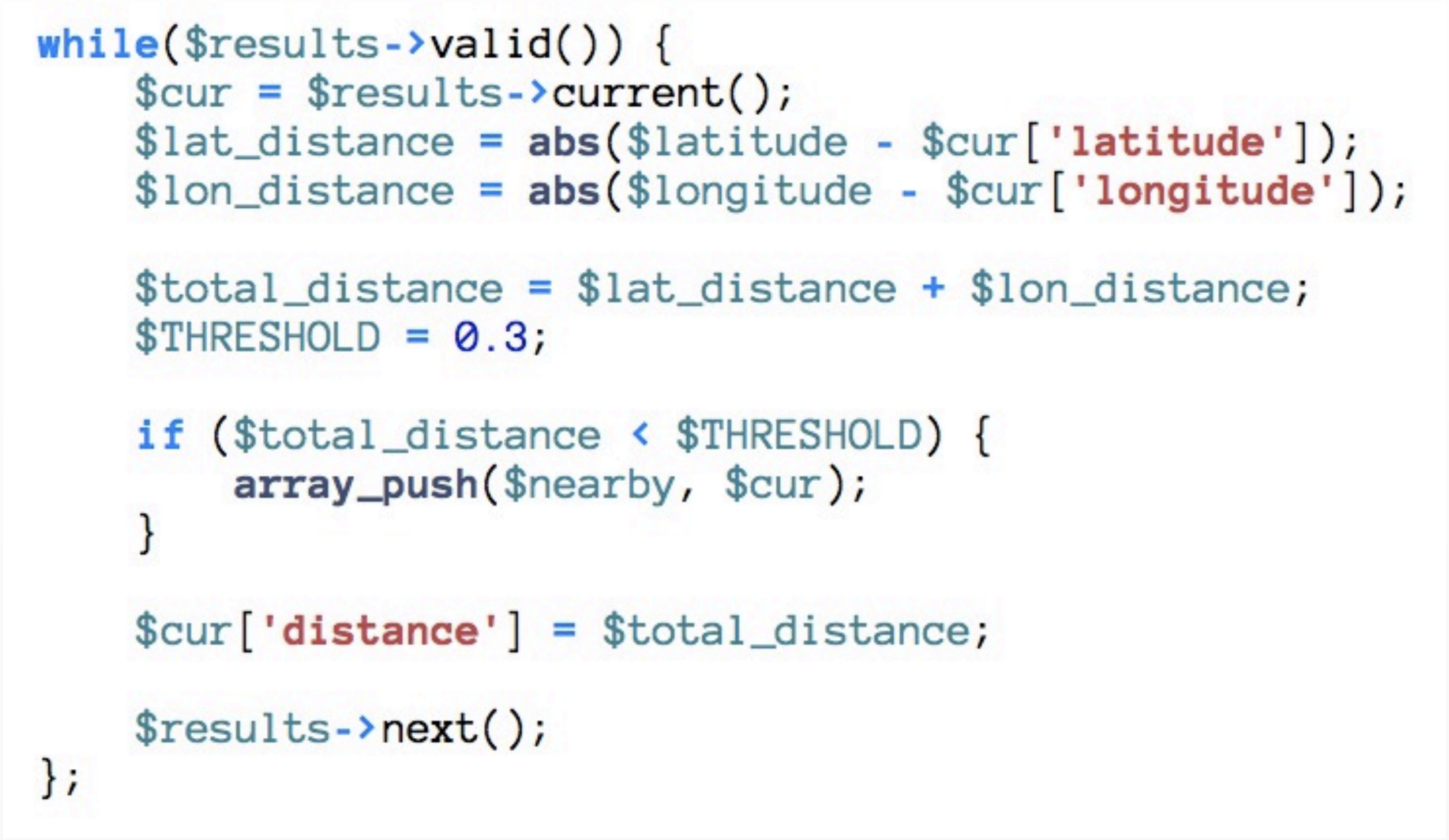
Grab all checkins

```
// in real app we would search in the SELECT
```



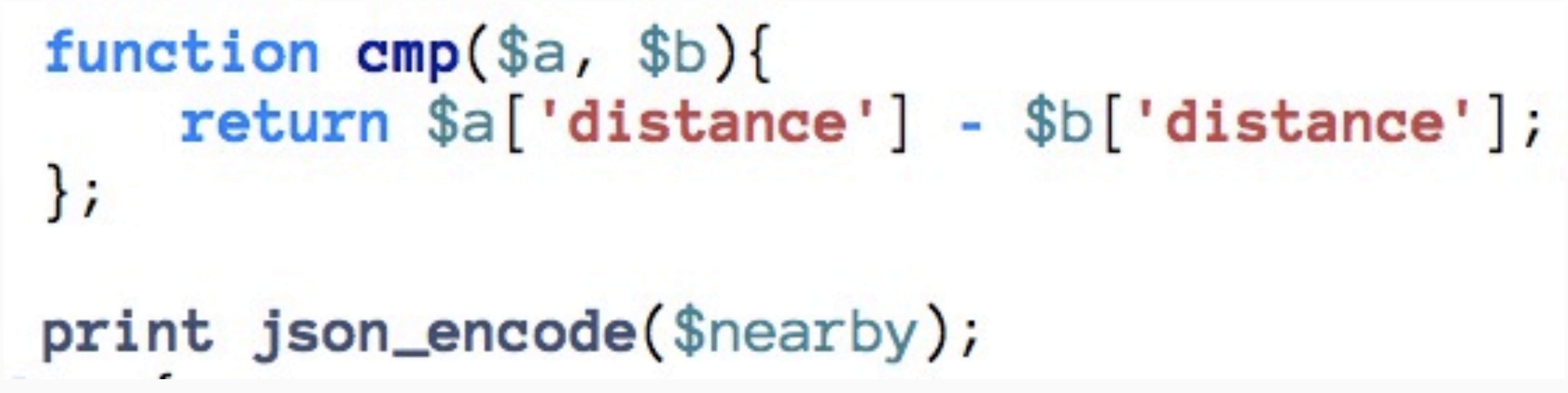
```
$command = "SELECT * FROM checkins";

$results = $db->query($command);

$nearby = array();
```

Find nearby



```
while($results->valid()) {
    $cur = $results->current();
    $lat_distance = abs($latitude - $cur['latitude']);
    $lon_distance = abs($longitude - $cur['longitude']);

    $total_distance = $lat_distance + $lon_distance;
    $THRESHOLD = 0.3;

    if ($total_distance < $THRESHOLD) {
        array_push($nearby, $cur);
    }

    $cur['distance'] = $total_distance;

    $results->next();
};
```

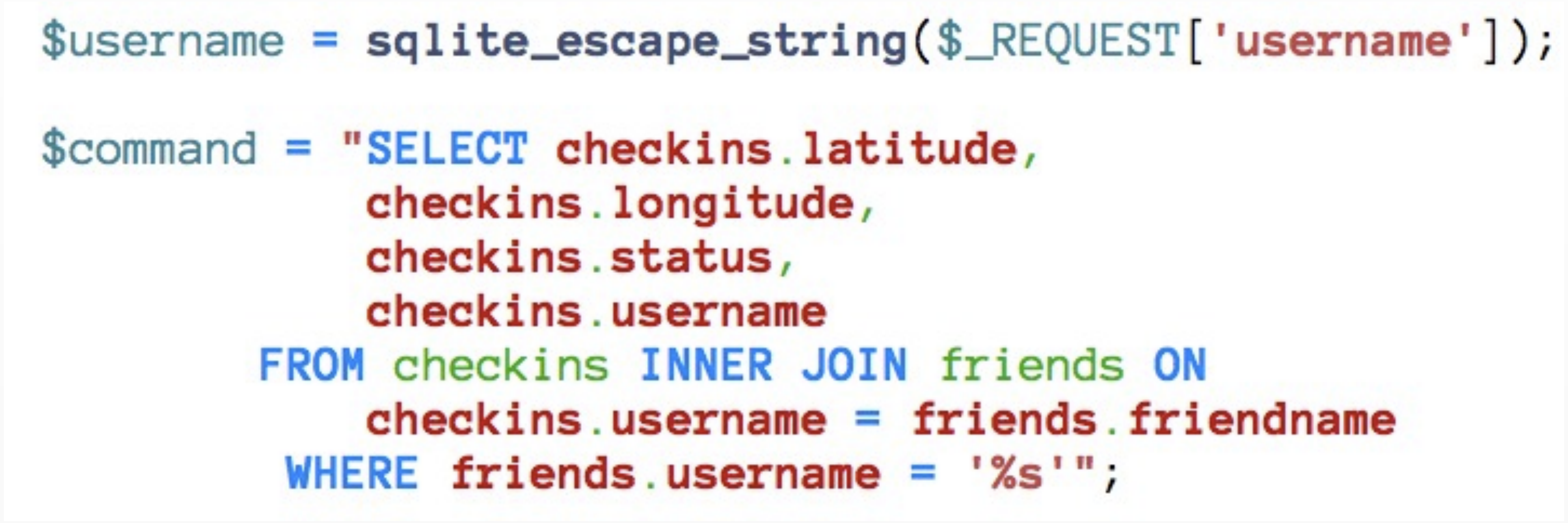
Sort & Print



```
function cmp($a, $b){
    return $a['distance'] - $b['distance'];
};

print json_encode($nearby);
```

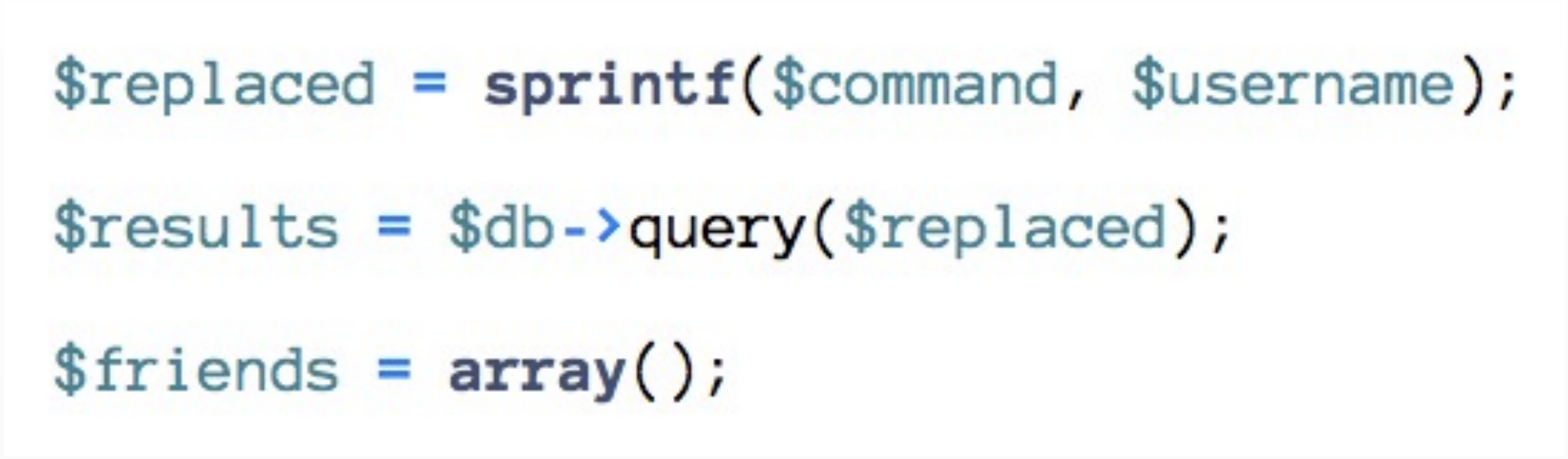
friends.php

```
// JOIN
```

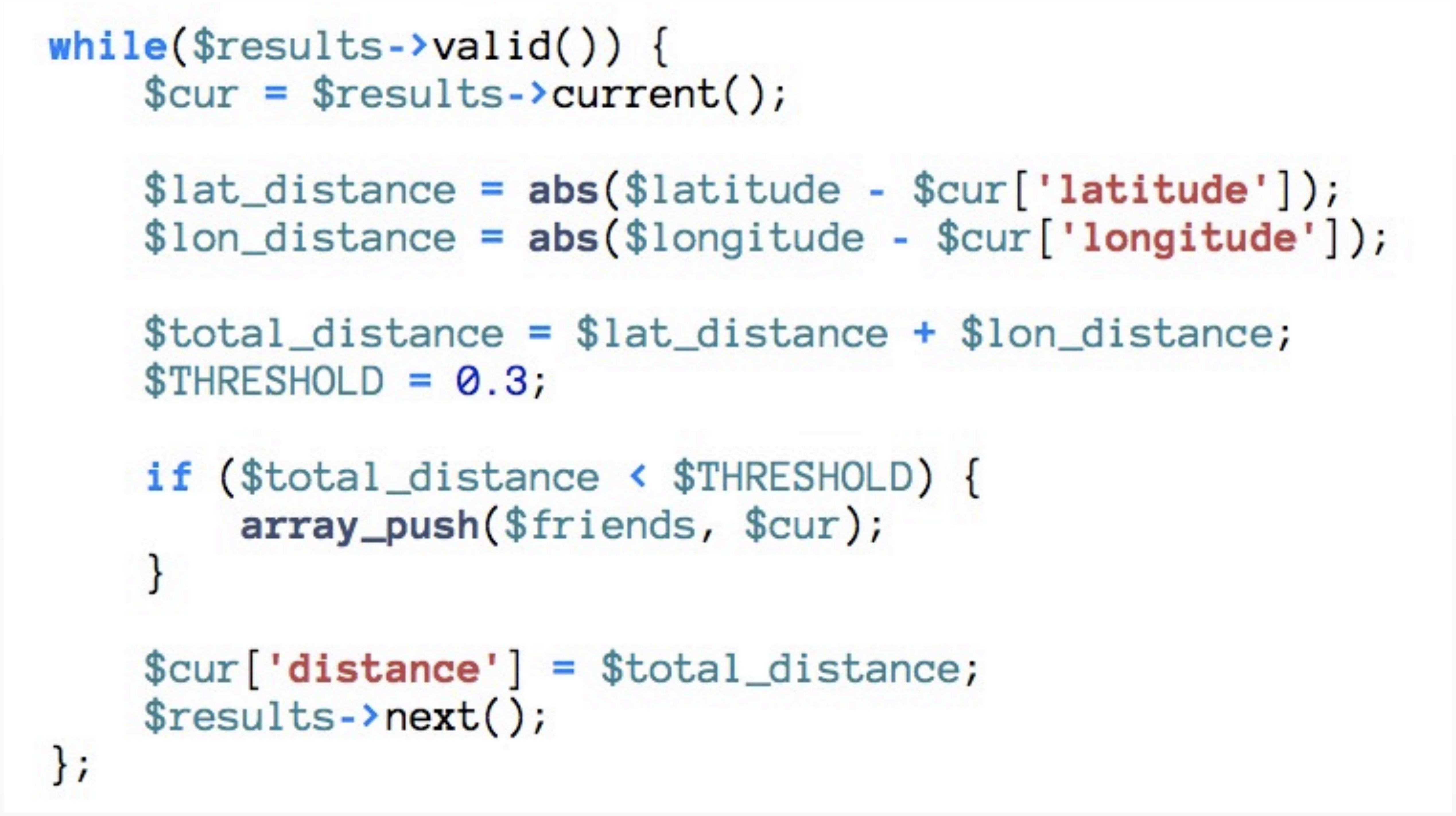


```
$username = sqlite_escape_string($_REQUEST['username']);

$command = "SELECT checkins.latitude,
            checkins.longitude,
            checkins.status,
            checkins.username
        FROM checkins INNER JOIN friends ON
            checkins.username = friends.friendname
         WHERE friends.username = '%s'";
```

Run command

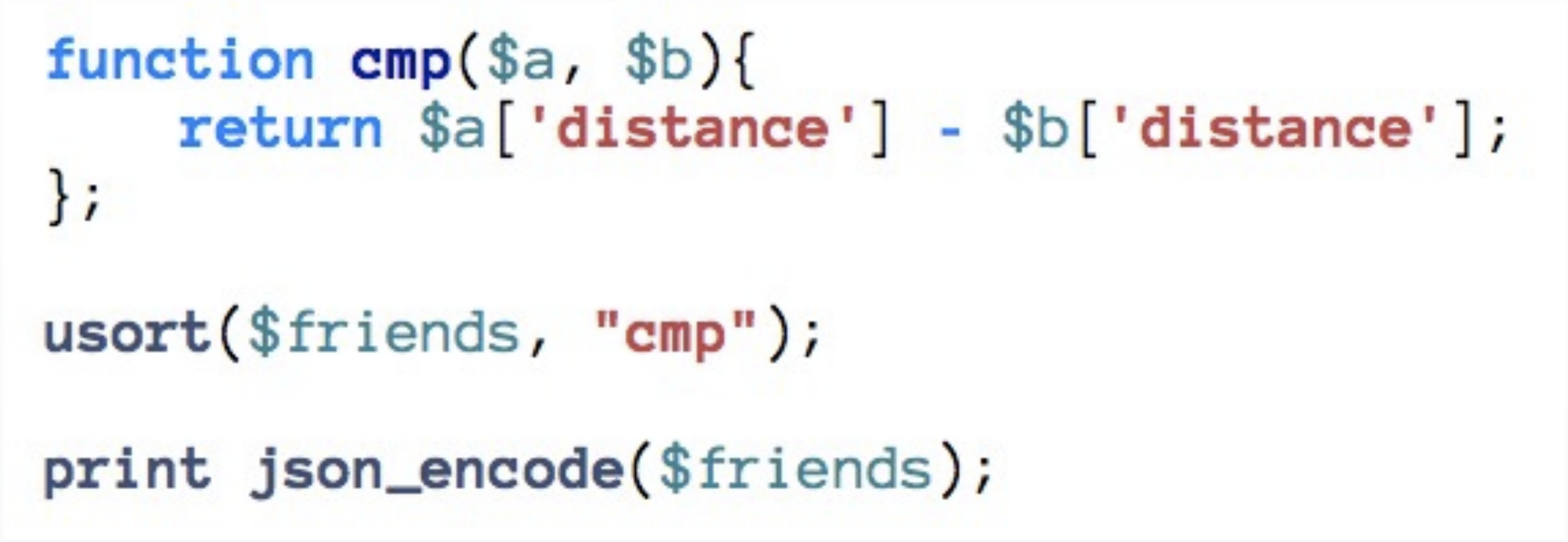

```
$replaced = sprintf($command, $username);

$results = $db->query($replaced);

$friends = array();
```

Iterate over results



```php
while($results->valid()) {
    $cur = $results->current();

    $lat_distance = abs($latitude - $cur['latitude']);
    $lon_distance = abs($longitude - $cur['longitude']);

    $total_distance = $lat_distance + $lon_distance;
    $THRESHOLD = 0.3;

    if ($total_distance < $THRESHOLD) {
        array_push($friends, $cur);
    }

    $cur['distance'] = $total_distance;
    $results->next();
};
```

Sort & Print



```
function cmp($a, $b){
    return $a['distance'] - $b['distance'];
};

usort($friends, "cmp");

print json_encode($friends);
```

Closing the loop

friends-9.html

- .get -> .getJSON
- Remove placeholder data

Final Demo

Q's?