Client-side Programs

Until now we've written server-side programs, which return output to the client.

With client-side programs, the program is sent to the user and run on the client computer.

The client browser must understand the language that the program is written in, and we've chosen JavaScript because it's the only language that all the browsers really understand.

Hello World

greetings.js:

```
var now = new Date();
window.alert('Hello world. It is ' + now + ', right now.');
```

page.html must include a `<script>` tag in the header:

```
<script src="greetings.js"></script>
```

When the client requests the webpage, the program will be returned with the page, and then run. A popup will appear with today's date.

Note: concerning the <script> tag

You can include JavaScript code in the script tag (between the <script> and </script> tags), but it's bad practice.

JavaScript

- Designed for writing programs embedded within other software applications.
- Core language is very standard, should be very familiar.
- Client-side JavaScript[1] includes objects to:
    - Control the browser
    - Interact with the user
    - Communicate with the server
    - Alter the document content
- There are server-side extensions, and others.
- JavaScript and Java are both partly inspired by C but are not related.

HTML Canvas

3 Files:

- A HTML web page, particles.html
- A CSS stylesheet, particles.css
- A JS program, particles.js

We use a `<canvas></canvas>` tag, with nothing between the tags, and specify a height and width.

On the CSS

To get the canvas in the centre, we put a margin on the left and on the right, and set them both to auto. Then the browser sorts itself out.

The JavaScript

The initial and ending lines ensure that variables in different files with the same name won't clash.

We select the canvas with the document.querySelector method, and we create a 'context', which will be how we interact with the canvas, like using a cursor for the database.

Declaring variables

Variables in JavaScript should be explicitly declared at the beginning:

```
var hourly_pay
var hours_worked

hourly_pay = 10
hours_worked = 30
```

This can also be done with the initial assignment, as a shortcut:

```
var hourly_pay = 10
…
```

---

1. These client-side extensions are less standardised, and support can differ from browser to browser.↩