Client-side computation

Validation

- Server-side validation essential
  - No guarantees that request actually came from a given front-end!
- But some client-side validation can reduce hits on server
- Example: email, date range, sanitization (no invalid characters) etc.
- Similar validation to backend, but now in front-end script
- Extra work, but better user experience

https://developer.mozilla.org/en-US/docs/Learn/Forms/Form_validation

Inbuilt HTML5 form controls

- Partial validation added by HTML5 standard
- `required`: mandatory field
- `minlength`, `maxlength`: for text fields
- `min`, `max`: for numeric values
- `type`: for some specific predefined types
- `pattern`: regular expression pattern match

Important: older browsers may not support all features.

Is backward compatibility essential for your app?

JavaScript validation

Constraint Validation API:

https://developer.mozilla.org/en-US/docs/Web/API/Constraint_validation

- Supported by most browsers
- Much more complex validation possible

Remember: not a substitute for server-side validation!

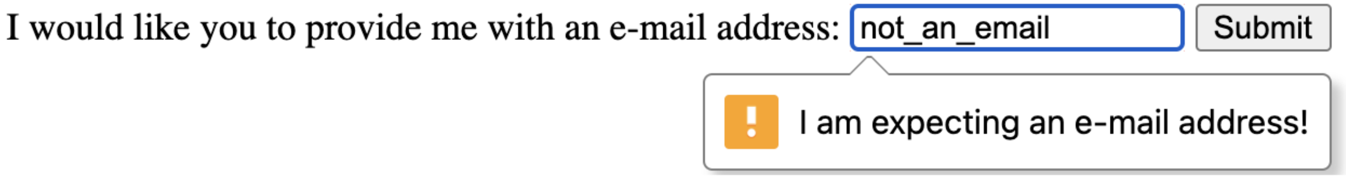
```html
<form>
  <label for="mail">I would like you to provide me with an e-mail address:</label>
  <input type="email" id="mail" name="mail">
  <button>Submit</button>
</form>
```

```javascript
const email = document.getElementById("mail");

email.addEventListener("input", function (event) {
  if (email.validity.typeMismatch) {
    email.setCustomValidity("I am expecting an e-mail address!");
  } else {
    email.setCustomValidity("");
  }
});
```

```
const email = document.getElementById("mail");

email.addEventListener("input", function (event) {
  if (email.validity.typeMismatch) {
    email.setCustomValidity("I am expecting an e-mail address!");
  } else {
    email.setCustomValidity("");
  }
});
```


I would like you to provide me with an e-mail address: not_an_email

I am expecting an e-mail address!

Captcha

- Problem: scripts that try to automate web-pages
- Can generate large number of requests in short time - server load
- Railway Tatkal, CoWin appointments etc.

Solution

- Prove that you are a human
- Limited number of clicks possible per unit time
- Script on page will generate some token - server will reject requests without the token

Crypto-mining ?

- Javascript is a “complete” language
- Can implement any computation with Javascript
- Modern JS engines very powerful, fast
    - Can even access system graphics processor (GPU) for rendering etc.
- Run a simple page that loads and runs a JS script
- Script will send results back to server through async calls
- Client may not even be aware!

Security Implications

Sandboxing

- Should JS be run automatically on every page?
  - Yes: provides significant capabilities
  - No: what if the page tries to load local files and send them out to server?
- Sandbox: secure area that JS engine has access to
- Cannot access files, network resources, local storage
- Similar to a Virtual Machine, but at higher level (JS interpreter)

Overload and DoS

- DoS: Denial of Service
- Run a script that takes over the browser engine and runs at high load
- Difficult to even navigate away from the page, or close the page
- Potentially exploit bugs in browser
- Server attack:
    - Replace some popular JS file with a bad version
    - Will be loaded by large number of sites, users - can write script to access some other site
    - Target site will be hit by huge number of requests from several sources, very difficult to control

Access to native resources

- Can JS be used to write fully native applications?
- Access to resources like local storage, sensors (tilt, magneto, camera?)
- Can be permitted explicitly through browser

Can also be compiled directly to native resources!

- Reduce browser overheads
- Smoother interaction with system

Summary

- Frontend experience determined by browser capabilities
    - Basic HTML + CSS rendering - styling
    - Javascript / client-side scripting for user interaction, smoother integration
- Native clients possible
- Potentially serious security implications!
- Always validate data again at server, do not assume client validation
    - HTTP is stateless: server cannot assume client was in a particular state!