Progressive Enhancement

A design/architecture philosophy

- Base content should be accessible to all browsers
    - Semantics!
    - No JS
- Extra/Altered layout with sophisticated CSS
- Extra/Altered functionality with JS

Benefits of Progressive Enhancement

- Available to widest possible audience
    - Not everyone is running latest browsers
        - Not everyone CAN run latest browsers
    - Including unknown audiences of the future
    - Such as programs that can't run JS
        - or can't understand visual effects
        - Existing and future
- Potential flexibility when faced with change
    - Strong separation of content and presentation

Graceful Degradation

- Flipside of Progressive Enhancement
    - Same goal though: Maximize availability
- Write for modern browsers
    - Fail back to older standards
        - CSS
            - if invalid property/rule, ignore it
        - JS
            - Feature detection

Progressive Enhancement: Basic How To

- Have the page be usable
    - With no CSS
    - With no JS
    - Requires Semantic HTML
    - Requires meaningful text order
    - Requires working forms/links
- Next, if CSS loads, page is nicer
- Next, if JS loads, page is nicer
    - Ex: Adds form validation
    - Ex: e.preventDefault and changes content

Cons and Costs of Progressive Enhancement/Graceful Degradation

- Extra Development effort
  - Smaller audience impact
    - No audience impact?
- Extra Testing effort
  - Have and use old browsers?
    - Other Operating Systems?
- Many like the idea, relatively few implement
  - Big companies
  - Focused publishers

Use of Progressive Enhancement is not a binary

- Can use partially
    - Particularly when adopting newer features
- Can create MVP (Minimal Viable Product)
    - Make sure core features work for all
    - Nicest options for latest
    - May be done as separate app
        - Ex: Gmail "Basic HTML" option