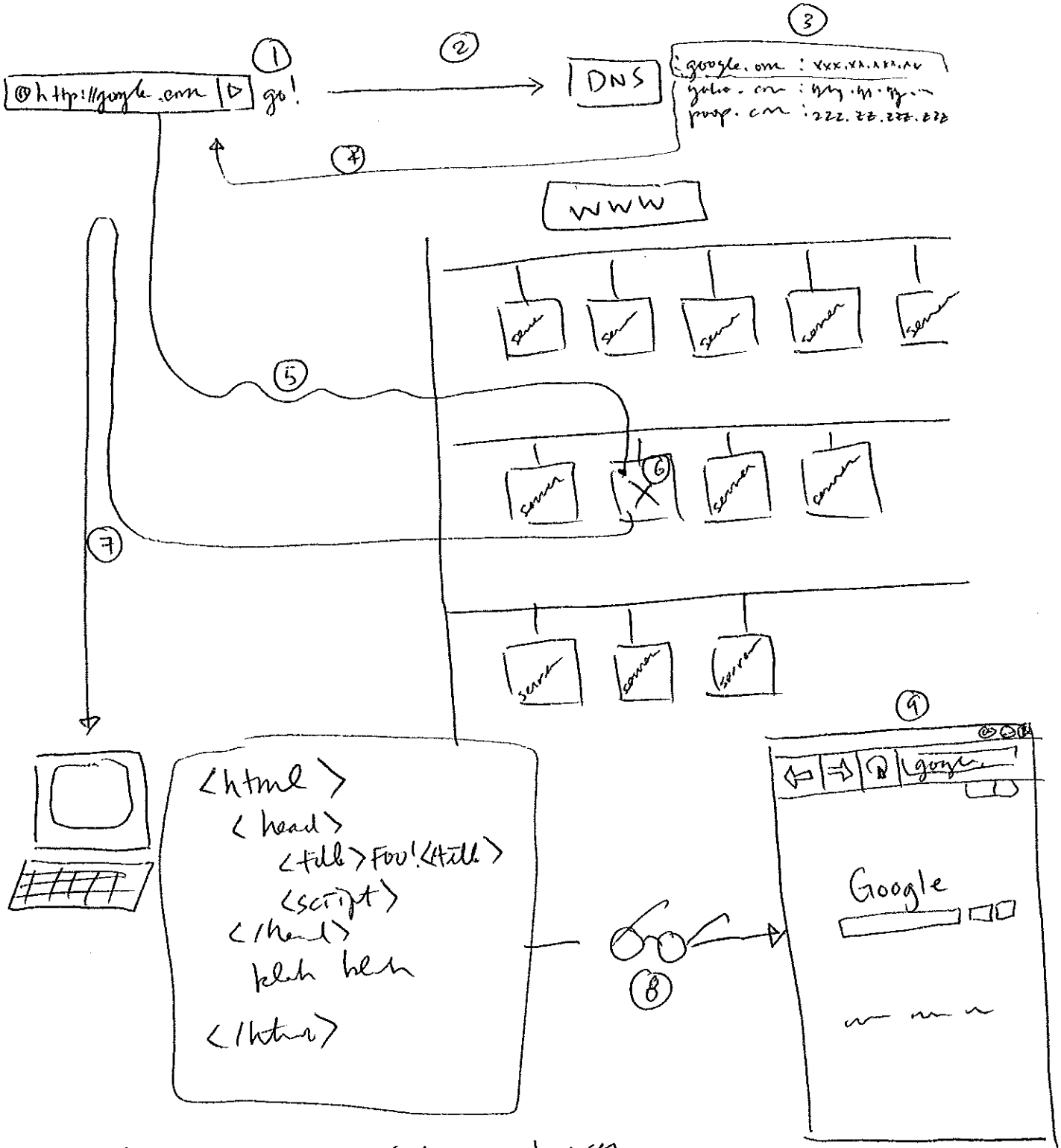
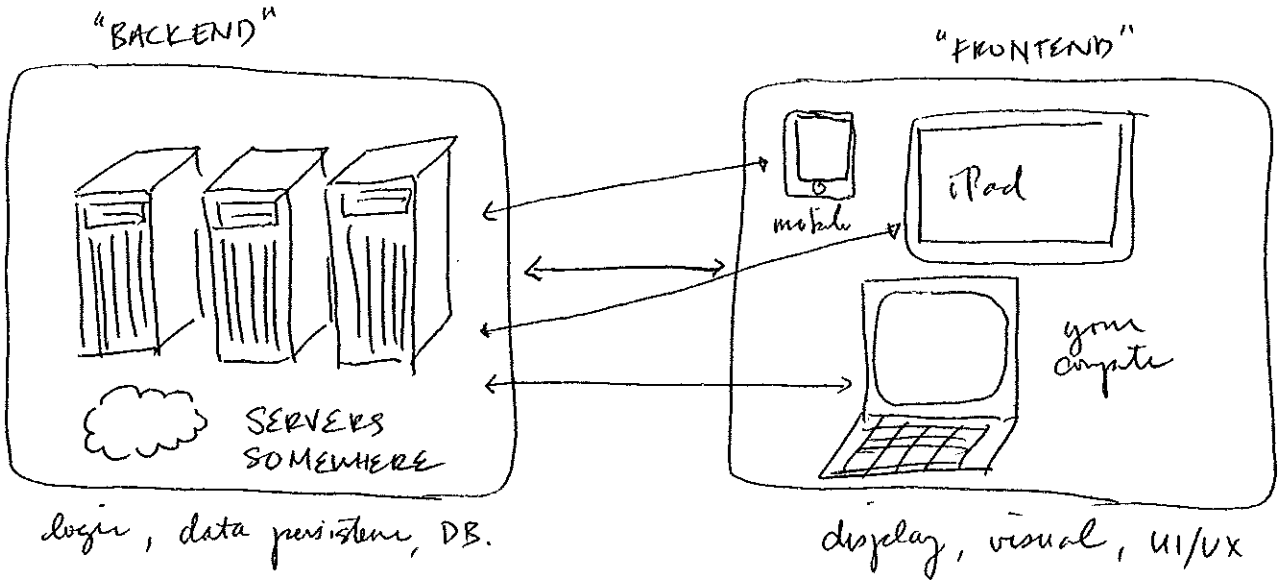
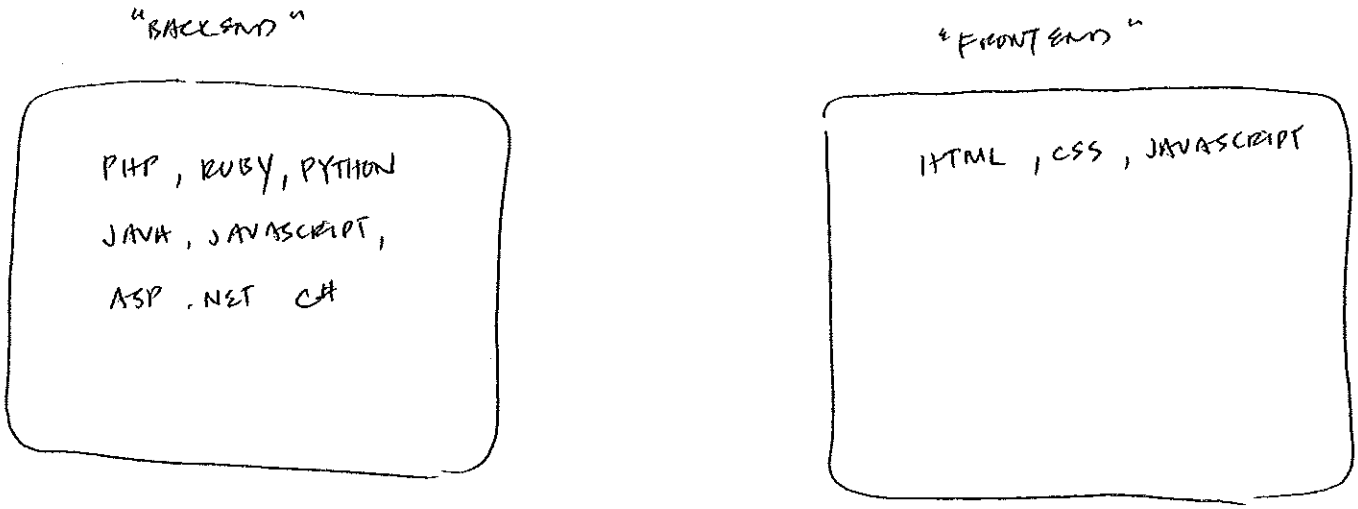


① type "google.com" into yur browser.
② send request to DNS
③ look up "addrss" of "google.com"
④ send it back.
⑤ send request to addrss
⑥ server process request "BACKEND"
⑦ return a response (code)
⑧ filter/interpret response. "FRONTEND"
⑨ display to user

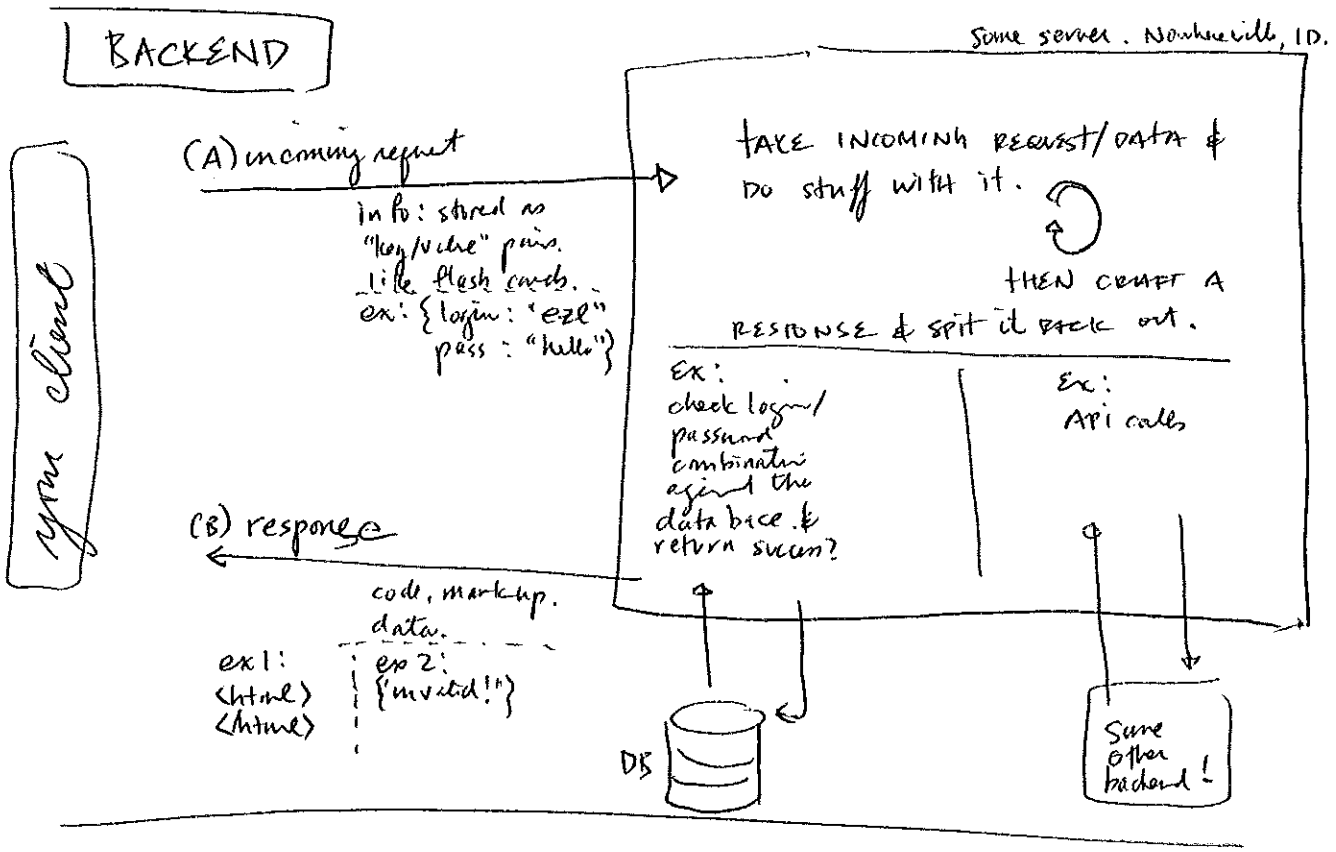

"BACKEND"
"FRONTEND"
SERVERS SOMEWHERE
iPad
your computer
login, data persistence, DB.
display, visual, UI/UX

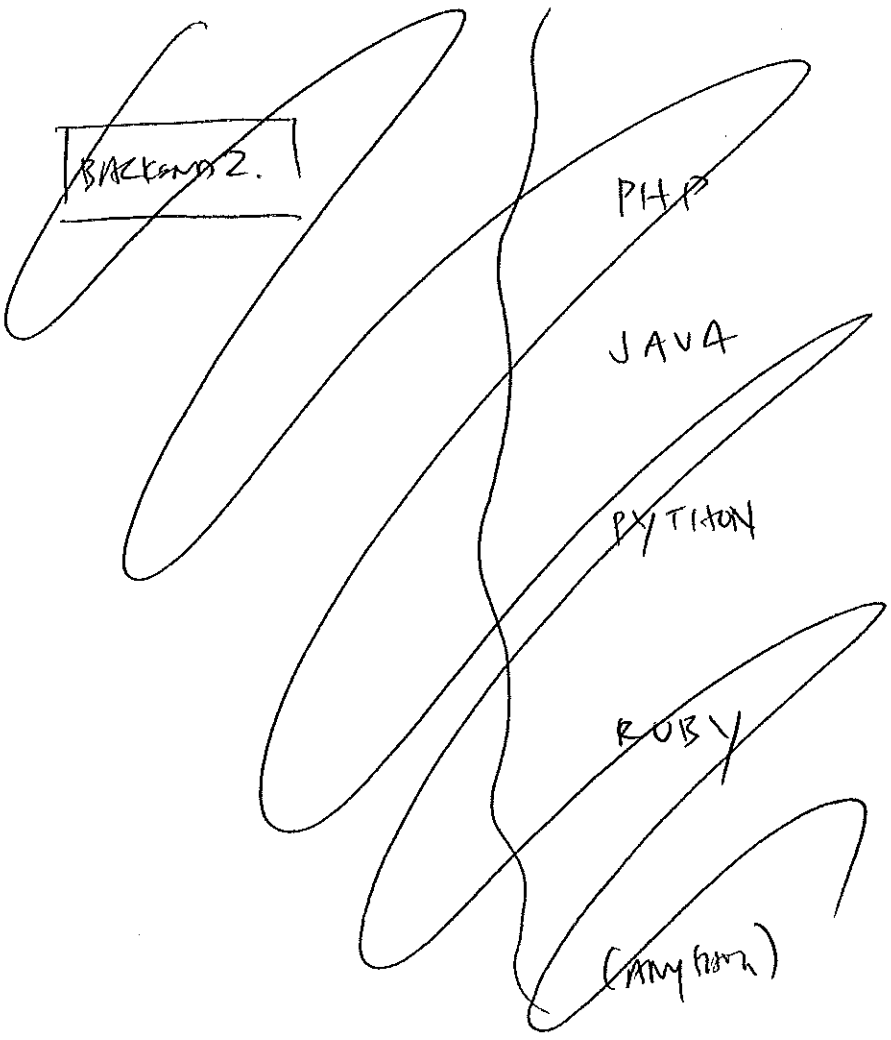

"BACKEND"
PHP, RUBY, PYTHON
JAVA, JAVASCRIPT,
ASP .NET C#
"FRONTEND"
HTML, CSS, JAVASCRIPT

HOUSE ANALOGY

BACKEND is STUFF YOU BUILD THE STRUCTURE WITH, THAT MAKES YOUR HOUSE FUNCTIONAL. WALLS, LOAD BEARING SUPPORTS, ELECTRICAL, PLUMBING, ETC

FRONTEND is AESTHETICS, PAINT COLOR, LIGHT FIXTURES, FRENCH DOORS, CROWN MOULDING.


BACKEND
Some server. Nowhere ID, ID.
your client
(A) incoming request
info: stored as "key/value" pairs. like flash cards.
ex: {login: "ezl" pass: "hello"}
TAKE INCOMING REQUEST/DATA & DO stuff WITH it.
THEN CRAFT A RESPONSE & SPIT it BACK out.
EX: check login/password combination against the database & return success?
Ex: API calls
(B) response
code, markup, data.
ex 1: <html> <html>
ex 2: {"invalid!"}
DB
Some other backend!


BACKEND 2.
PHP
JAVA
PYTHON
RUBY
(ANYTHING)

FRONTEND

server response

<html>

...

<h1>bill</h1>

<p> <p>

.class{

color: red

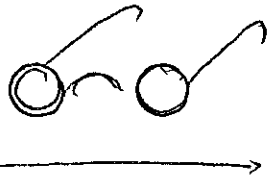
}

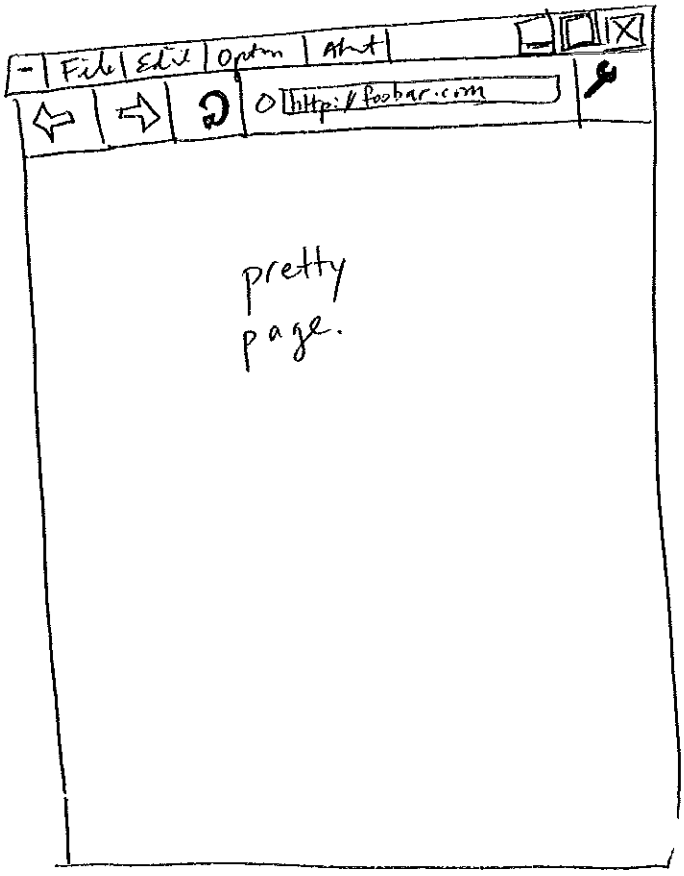
$.(javascript)

blah blah

code markup

jsm

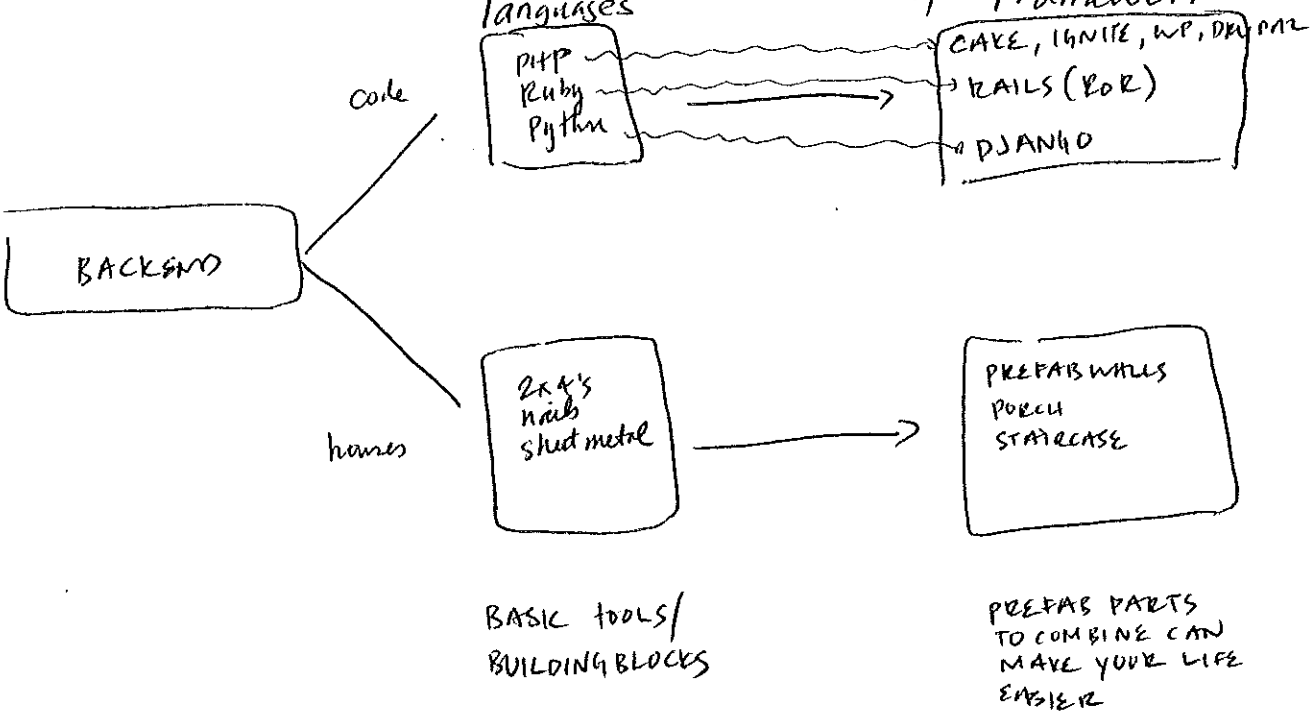




ex: Firefox

YOUR "CLIENT" IS A FILTER/LENS THAT TAKES THE SERVER'S RESPONSE & MAKES IT SENSIBLE FOR YOU

OK... LETS SEE HOW FAR WE CAN STRETCH THIS HOUSE ANALOGY... framework



use these → PHP, PYTHON, RUBY, } ARE **LANGUAGES**

don't use these → (ASP, .NET, JAVA, C#, C++, ERLANG, SCALA,) } ARE **LANGUAGES**

They are substitutes for one another.

RUBY ON RAILS
DJANGO
CAKE PHP, IGNITE } ARE **FRAMEWORKS**

They are collections of tools combined that make constructing commonly desired features easy.

WARNING! HOUSE ANALOGY is
GONNA GET REALLY FLIMSY HERE...

HTML

hypertext markup language

DEFINES THE BASIC STRUCTURE OF YOUR PAGE & CAN CONTAIN INFORMATION ABOUT WHAT THE CONTENT REPRESENTS (paragraphs, headings, lists, links, quotes, definitions, etc)

IT LOOKS LIKE A SERIES OF TAGS WITH CONTENT:

```
<h1> this is a header </h1>
<p>
<form>
<img src="..." />
```

CSS

cascading style sheets

DEFINES APPEARANCE, LOOK, & LAYOUT OF PAGE CONTENT

IT LOOKS LIKE

```
#elementid {
    color: red;
}
.right-justify {
    float: right
}
```

JAVASCRIPT

PROVIDES INTERACTIVITY TO WEB PAGES

IT LOOKS LIKE

(changed my mind... you shouldn't care)

flimsy housing analogies

HTML

ARRANGEMENT of FURNITURE INSIDE YOUR HOUSE / ROOM

CSS

PAINT, CURTAIN SELECTION, STYLE of FURNITURE

JAVASCRIPT

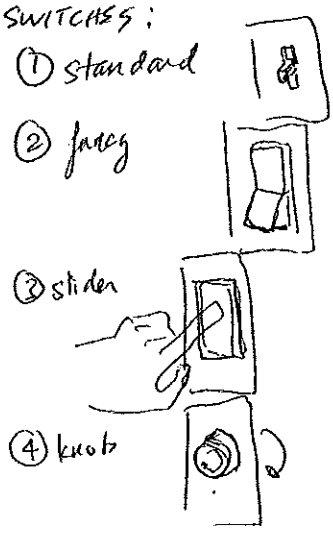
HOW YOU INTERACT W/ THE SPACE.
FOR EXAMPLE, LIGHT SWITCHES:

① standard

② fancy

③ slider

④ knob



REMEMBER HOW WE DESCRIBED
PHP / PYTHON / RUBY ≈ SUBSTITUTES ?

HTML / CSS / JAVASCRIPT ARE COMPLEMENTS.
ONE CAN NOT REPLACE THE OTHER.

JAVASCRIPT IS NOT
THE SAME AS JAVA.

YOU DON'T CARE ABOUT
JAVA.

IN REAL LIFE, THE DISTINCTION BETWEEN FRONT/BACK END ISN'T ALWAYS VERY CLEAR CUT & MANY ENGINEERS ARE BETTER/WORSE @ VARYING PARTS OF THE STACK.