Tomorrow's tours...

Welcome!

- Early morning tour here
- Leave here @ 9:30 am
- 10:30-11am arrival

The Group: 25 people

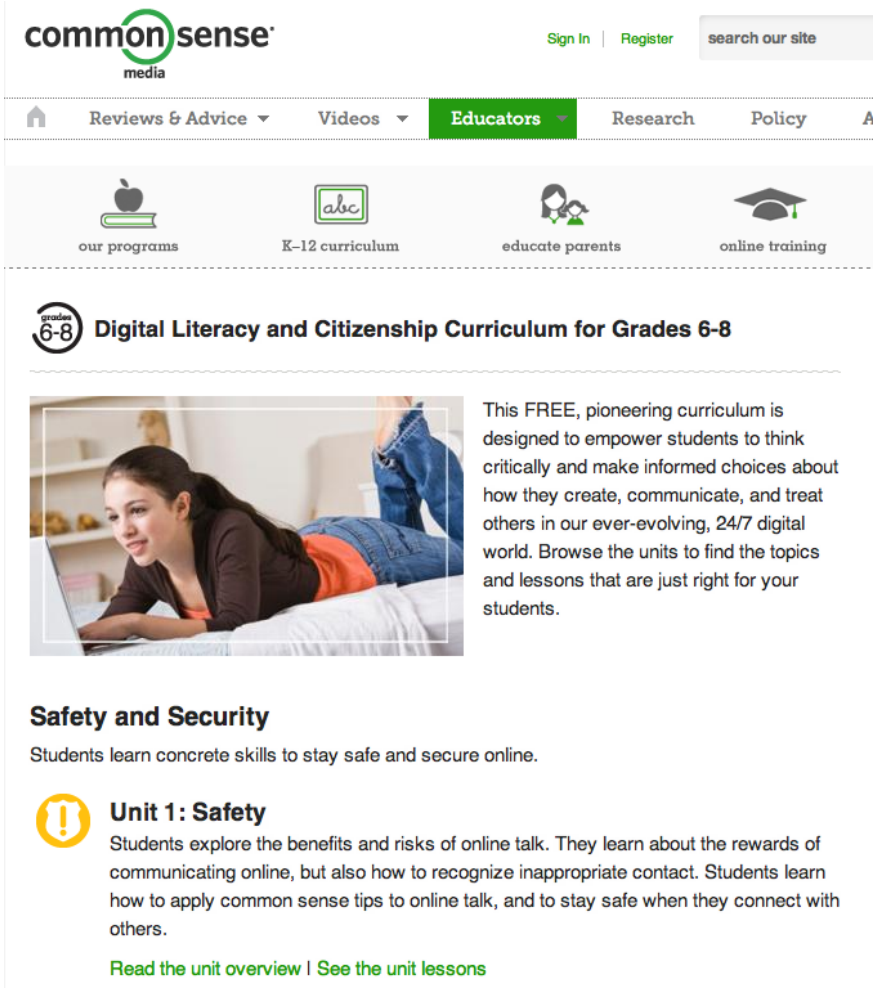
Agenda: Friday, July 13, 2012
10:30am - Arrive: Karen will have parking passes f
how many you'll require
11:00am - Tech Talk
11:45am - Google Panel Discussion
12:30pm - Lunch
1:30pm - Tour
2:30pm - Depart

Social Responsibility on the Web

Day 4, Session 1

Common Sense Media has an extensive, free and tested set of lessons aimed at middle schoolers.

(These are the Social-Responsibility unit of MyCS)



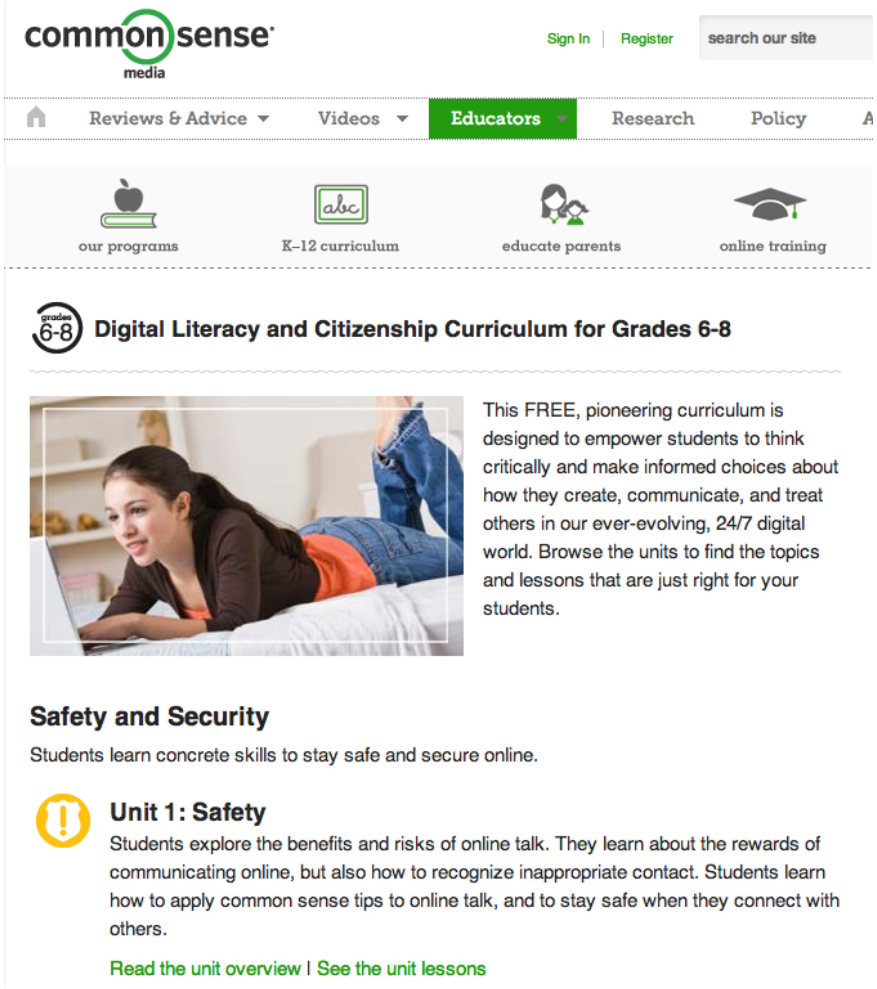
9 units for grades 6-8

There are materials for
K-5
6-8
9-12

... and they're matched to national technology standards for all of these groups: (i) ELA, (ii) AASL, (iii) ISTE

Common Sense Media has an extensive, free and tested set of lessons aimed at middle schoolers.

(These are the Social-Responsibility unit of MyCS)



9 units for grades 6-8

(1) Online safety
(2) Security
(3) Digital life
(4) Privacy and "digital footprints"
(5) The connected culture
(6) Self-expression and identity
(7) Respecting creative work
(8) Searching well
(9) Research and evaluation

At least five weeks' worth of plans...

Life Without The Internet?

Thought experiment:

- Two ways life would be better/worse without cell phones, the internet, and other modern technology?

better	worse
1)	1)
2)	2)

- Could you live a week without a cell phone, a computer, etc.?

Can You Trust Websites?

Discuss if you would trust websites you found because:

- a friend linked you to it
- a different website linked you to it
- a teacher or parent told you to use it

Can You Trust Websites?

How do ***you*** decide whether
to believe a website?

Try these first:

- http://en.wikipedia.org/wiki/Main_Page
- http://www.dhmo.org/facts.html
- http://www.cdc.gov/phpr/zombies.htm
- http://www.cnn.com/
- http://snarxiv.org/

RESEARCH AND EVALUATION >
IDENTIFYING HIGH-QUALITY SITES

STUDENT HANDOUT

Test Before You Trust

grades 6-8

Group Members _____

Date _____

Name of Site _____

URL _____

Website Test

Purpose of the Site	Circle one		Add details to explain
1. Can you tell if the site is fact or opinion? (If the information seems one-sided, or biased, you will have to go elsewhere to hear the other side of the issue.)	YES	NO	
2. Is the site free of advertising?	YES	NO	
3. If there are ads, is it easy to tell the difference between ads and content?	YES	NO	
4. Is the site sponsored by any organizations?	YES	NO	
5. Is it clear who the site is for? (for example, college students or young children)	YES	NO	
6. Is the tone calm and fair? (Sites that are mean and angry may not be good sources of information.)	YES	NO	

Common Sense Media's checklist (these are the first 6 of 18 criteria...)

Cyber Bullying

What is it?

The use of digital media tools such as the Internet and cell phones to deliberately upset or harass someone else.

What makes ***cyber bullying*** different from ***bullying*** ?

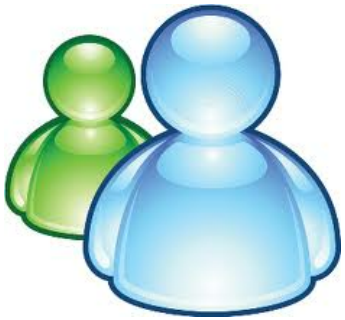
Cyber Bullying vs. Bullying

- Kids often use ***more hurtful and extreme language*** online.
- Cyber bullying can happen ***24/7***.
- Cyber bullying is ***often very public***.
- Cyber bullying can be ***anonymous***.
- The age and size of a person are often less important with cyber bullying because people are ***not face to face***.

Handling Cyber Bullying

- ***Sign off*** the computer.
- ***Don't respond*** or retaliate.
- ***Block*** the bully from messaging you.
- ***Save the messages*** in case the bullying does not stop.
- ***Talk*** to a friend or trusted adult.

Safe Online Talk



Be sure to ask yourself the following questions:

- Has this person asked me to keep any information ***secret*** ?
- Has this person flirted with me or asked me about anything ***sexual*** ?
- Has this person asked me about anything ***private***?
- Have I felt ***pressured*** by this person to do anything?
- Do I feel like I'm sticking to my ***values*** when talking to this person?

CSM's lesson: *it's not always obvious!*

CONNECTED CULTURE > CHART IT

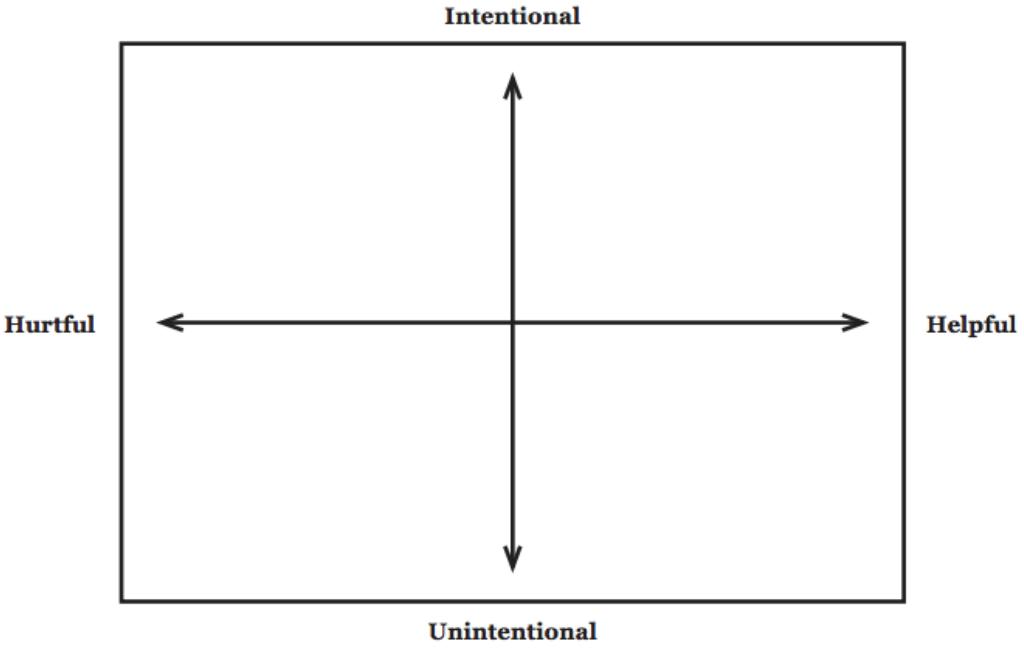
You Chart It

STUDENT HANDOUT grades 7-8

Name(s) **Class** **Date**

Directions

Read the stories below about different types of online relationships. For each of these situations, take the point of view of the person in bold. Imagine a point on the y axis that stands for how intentional this person's actions are. Then imagine a point on the x axis that stands for how helpful or hurtful that person's actions are. Now imagine lines that stem from the two dots meeting in the middle. Place a dot ther



Plot 8 scenarios on this chart...

(see handout)

Respecting Creative Work Online

ASK.

- How does the author or artist say I can use the work?
- Do I have to get the creator's permission first?

ACKNOWLEDGE.

- Did I give credit to the work I used?

ADD VALUE.

- Did I rework the material to make new meaning and add something original?

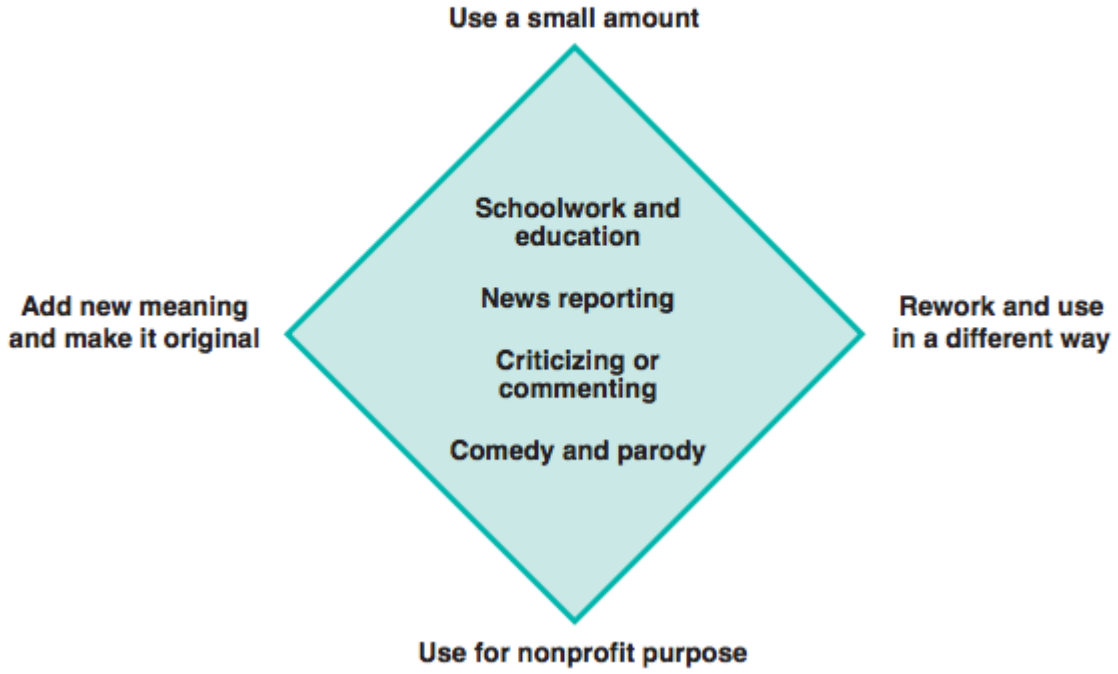
CSM's lesson: *four points of Fair Use*

RESPECTING CREATIVE WORK > A CREATOR'S RIGHT

The Four Point of Fair Use

STUDENT HANDOUT grades 6-8

FAIR USE: The ability to use copyrighted work without permission, but only in certain ways and in specific situations (schoolwork and education, news reporting, criticizing or commenting on something, and comedy/parody).



Try it for two examples...

CSM's lesson: *four points of Fair Use*

	Scary Mary	*DJ Earworm*
Fair-use points		
Unfair-use points		
Verdict		

HTML and CSS

Day 4, Session 2

What is HTML?

It is ***instructions*** to a web browser that tells it what is on a webpage.

HTML is mostly ***text***.

It includes special instructions ***similar to blocks*** in Scratch.

HTML stands for **HyperText Markup Language**

Where is HTML?

HTML is deciphered by browsers. Everything you see on a webpage comes from HTML.

To view it in any browser, you should be able to go to the menu and click ***View Source***.

Note that many websites *generate* their code, so don't be worried if you look at some HTML which seems quite unreadable.

A Paragraph Block

If we were in scratch, a **paragraph element** would look like this:

In HTML, since we don't have blocks, it looks like this:

```
<p>This is a paragraph!</p>
```

Opening and Closing Tags

We call ** ** and ** ** ***tags***.

** ** is an opening tag.
**** is also an opening tag.

** ** is a closing tag.
**** is also a closing tag.

Nesting Tags

Almost every open tag must have a close tag.

Tags can be **nested** just like blocks in Scratch.

Just like in Scratch, nested blocks must finish before the outer block.

```
<p>This is a <b>paragraph</b>!
</p>
```

Lots of Tags!

There are a lot of tags. A couple are:

** ** creates a paragraph.
**** creates **bold** text.
**** creates *italicized* text.
<h1></h1> creates a large header.

Don't worry about remembering these! Look there's a reference sheet!

Closing Tags?

Some tags do not require a close tag.

In all of these cases, it is because there is not **text** to put between tags.

**
** creates a line break.

<hr> creates a horizontal line.

<img src="picture.png"> displays an image.

Tags That Rely on Nesting

```
<ul>
    <li>Milk</li>
    <li>Crackers</li>
    <li>Broccoli</li>
</ul>
```

- Milk
- Crackers
- Broccoli

```
<ol>
    <li>Blue</li>
    <li>Green</li>
    <li>Orange</li>
</ol>
```

1. Blue
2. Green
3. Orange

Tags That Rely on Nesting

Tables are structured through **rows** and **cells**.

```
<table border="1">
    <tr>
        <th>Colors</th>
        <th>Shapes</th>
    </tr>
    <tr>
        <td>Red</td>
        <td>Circle</td>
    </tr>
</table>
```

Colors	Shapes
Red	Circle

A Simple Recipe Page

Go to http://jsfiddle.net/begerter/PrLCX/

Using HTML, add:

- a title as a **header**
- an **image** of the food the recipe creates
- images of the three most important ingredients
- a **list** of ingredients
- a **list** of instructions

Press 

to update the result. Press 

to save your recipe page.

Sample Recipe HTML

```
<h1>Spam Sandwiches!</h1>
<img src="https://encrypted-tbn2.google.com/images?q=tbn:
ANd9GcS_MHadc8QljRCFPc7M5gPzFnqB3jLKk_oH7M4DLUl5QrEmgXXuqw">
Ingredients:
<ul>
    <li>Spam</li>
    <li>Bread</li>
</ul>
<img src="https://encrypted-tbn0.google.com/images?q=tbn:
ANd9GcRvKWTjkP1QBVScqYaLjJgyUjrvTLgDMmIVUVG6HBJl6PMs8PBspg">
Instructions:
<ol>
    <li>Take out two slices of bread</li>
    <li>Spread spam on one of them</li>
    <li>Put other slice on top</li>
    <li>Enjoy!</li>
</ol>
```

What is CSS?

CSS stands for **Cascading Style Sheets**.

CSS lets you change how the HTML looks.

An **attribute** is something you can change with CSS.

Some attributes are color or size.

Using CSS

To use CSS, we write in the CSS panel of our jsFiddle:

We start by specifying the element we want to change.

```
p
{
    color:blue;
    font-size:20pt;
}
```

Then, within the area defined by the brackets, we give the **attribute name**, a **colon**, the **value** we want to give, and then a **semi-colon**.

RGB Values

RGB stands for **red green blue** which are the primary colors on computers.

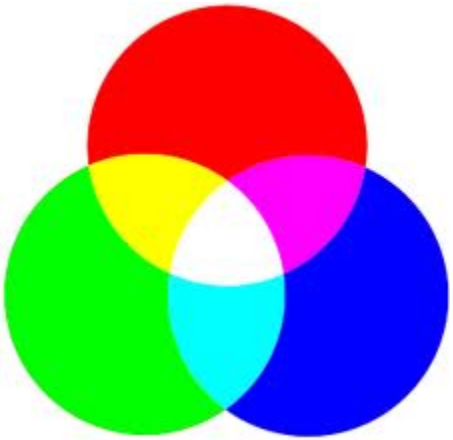
Whenever you want to use CSS to set an attribute such as **color**, you should use RGB notation:

```
color:rgb(100%, 0%, 0%);
```

Here, we have a *percentage* for how much red, green or blue we should include. **0%** means none of that color, and **100%** means all of that color.

RGB Worksheet

To get a better sense of how RGB values work (as opposed to the red, blue, yellow primary colors that happen in painting!), go through this worksheet with your partner.



The Tag

The ** ** tag just lets you structure the information on a HTML page more. It is often used nested:

```
<div>I am a div!
    <div> I am a div in a div!
</div>
</div>
```

The Problem

We have now seen that we can make all ** ** red, or all **<h1></h1>** really big.

But, what if we don't want *all* of the paragraphs to be red?

With CSS, you can **name** elements so you have more control over how everything is styled.

Naming

To name an element, you add an **id** as an attribute in the HTML.

Instead of your code being:

```
<p>I am a paragraph</p>
```

It will look like:

```
<p id="awesome">I am a paragraph</p>
```

Using Names in CSS

Then, to style that paragraph only, to **select** it in the CSS you would write:

```
#awesome
{
    color:rgb(100%, 0%, 0%);
}
```

Even if you say that anything with the ** ** tag should be blue, if you select a ** ** by a **name** and tell it to be red, it will be red, not blue.

Classes

Another way to **select** is by class. You can write:

```
<p class="myClass">I am a paragraph</p>
```

And, when you want to **select** it, you use:

```
.myClass
{
    color:rgb(0%, 0%, 100%);
}
```

Combining Selectors

Elements can even be **selected** by where they are.

The following header can be **selected** by giving information about what it is in, and then its own information:

p h1 or **#myId #otherId** or **.myClass .otherClass**

```
<p id="myId" class="myClass">
<h1 id="otherId" class="otherClass">
I am a paragraph and a heading
</h1>
</p>
```

Selecting Tips

Selecting by class is less important than **selecting** by name, but more important than just **selecting** by tag.

With **nested** HTML tags, if you give a style to the outer element, unless you also give the inner one a style, **the inner element will inherit the style of the outer element**.

Selection Practice

Predict and test different selectors with your partner on this worksheet.

More Recipes!

We've already created some HTML of a recipe. Now we'll add some CSS to make it pretty by following along with this worksheet.

Security

Day 4, Session 3

Security?

All information passed through the web can be seen by everyone (at least in theory).

How can a message get from Alice to Bob without Charley snooping it?

Let's introduce our participants...

Security?

All information passed through the web can be seen by everyone (at least in theory).

How can a message get from Alice to Bob without Charley snooping it?

Let's introduce our participants...

Security?

All information passed through the web can be seen by everyone (at least in theory).

How can a message get from Alice to Bob without Charley snooping it?

Let's introduce our participants...

Security?

All information passed through the web can be seen by everyone (at least in theory).

How can a message get from Alice to Bob without Charley snooping it?

Let's introduce our participants...

Security?

All information passed through the web can be seen by everyone (at least in theory).

If Alice and Bob share a secret (that Charlie doesn't know), they can use it to encrypt messages.

Security?

All information passed through the web can be seen by everyone (at least in theory).

The bits of Alice's message are ***encrypted*** by the bits of the shared secret ("the key").

Charley ***CAN SEE*** the message, but it's encrypted!

Because Bob has the shared secret bits ("the key"), Bob can ***decrypt*** the locked message

Creating a shared secret

But how can Alice and Bob create a shared secret?

Ideas?

Remember - Charley can see ***everything***!

Creating a shared secret

It's one of the ***nine algorithms***
that changed the future...

(1) Alice and Bob choose a public color (really, a number) -- *and everyone can see it*!

(2) Alice and Bob privately "mix" that public color with their own secret color and then they let *everyone see both mixes*!

(3a) Alice grabs Bob's mix -- and mixes in her color ***privately***!

(3b) Bob grabs Alice's mix -- and mixes in his color ***privately***!

(4) Alice and Bob now have the same color (really, a number) -- and *no one else knows what it is*!

Shared secret skit!

Creating a shared secret

In groups of three, choose an Alice, a Bob, and a Charlie. Alice and Bob use food coloring to create a shared secret; Charlie tries to figure it out!

(1) Alice and Bob choose a public color (really, a number) -- *and everyone can see it*!

(2) Alice and Bob privately "mix" that public color with their own secret color and then they let *everyone see both mixes*!

(3a) Alice grabs Bob's mix -- and mixes in her color ***privately***!

(3b) Bob grabs Alice's mix -- and mixes in his color ***privately***!

(4) Alice and Bob now have the same color (really, a number) -- and *no one else knows what it is*!

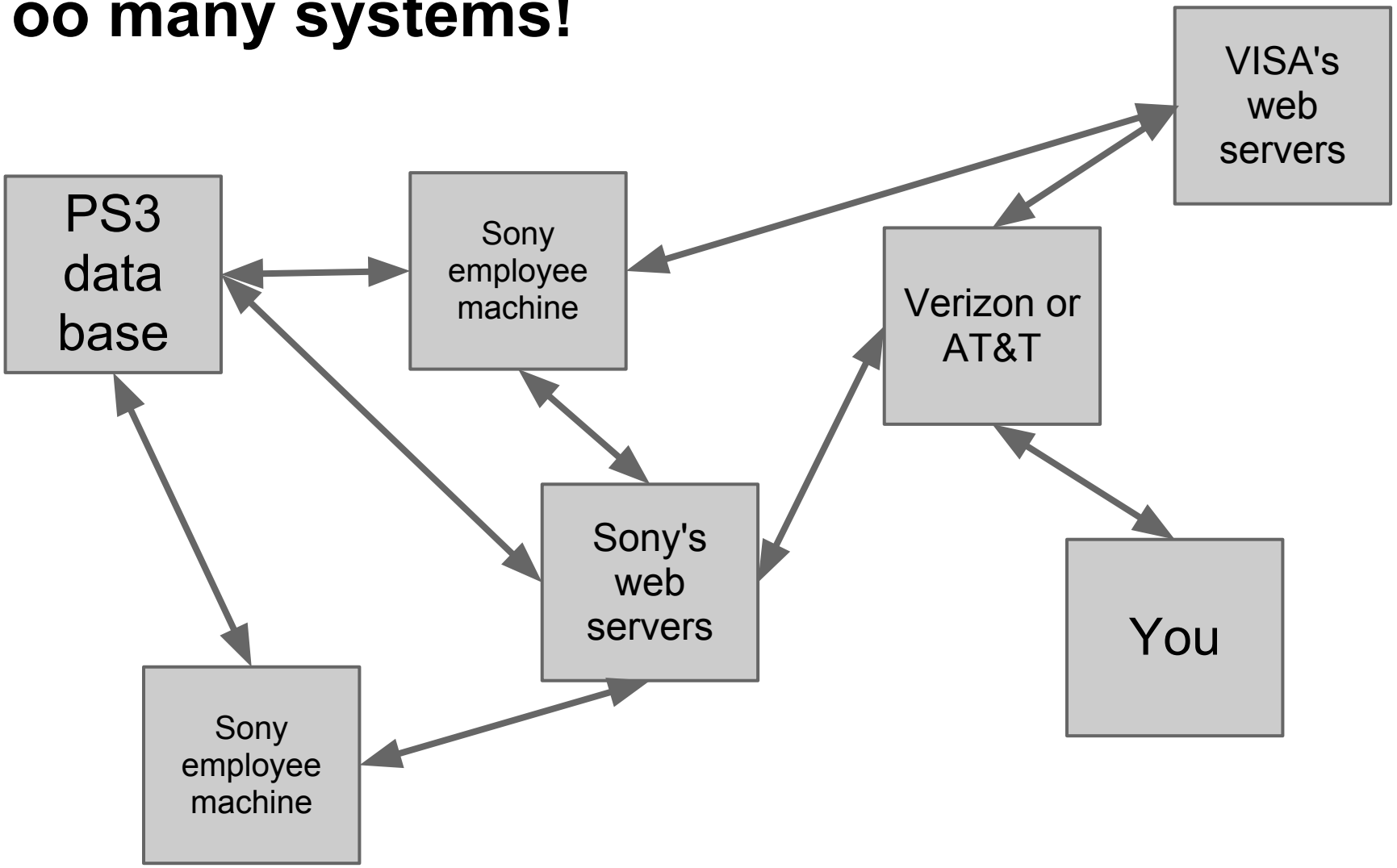
So, what are passwords for?

Passwords make sure you are
who you say you are!

Once your password is accepted, your computer creates a shared secret and can communicate securely.

But there are many, many systems that need to communicate with each other...

Too many systems!



What's the danger here?
Passwords, passwords, passwords!

CSM's password lesson plan

SECURITY > STRONG PASSWORDS

Password Challenge

STUDENT HANDOUT grades 6-8

Name(s) _____ **Class** _____ **Date** _____

Directions

You will create one strong and one weak password for an important historical figure. Both passwords should indicate something that is special or unique about that person. Use the "Do" tips from the **Password Tips Student Handout** to create the strong password, and use the "Don't" tips from the same handout to create the weak password.

Example for Abraham Lincoln:

Strong: 4score7yrs ("Four Score and Seven Years Ago ...")

Weak: HonestAbe

YOUR HISTORICAL FIGURE: _____

FACTS OR INFORMATION YOU LEARNED ABOUT YOUR HISTORICAL FIGURE: _____

_____

_____

STRONG PASSWORD: _____

WEAK PASSWORD: _____

See the "tips" page!

Javascript

Day 4, Session 4

What is Javascript?

Javascript is a **programming language**. Like most programming languages, it uses structures similar to Scratch's **ifs**, **fors**, **broadcasts**, **variables**, and **lists**.

Javascript, like most **programming languages**, is defined by **blocks of text**, like the HTML you have learned.

Javascript is the main **programming language** used on websites.

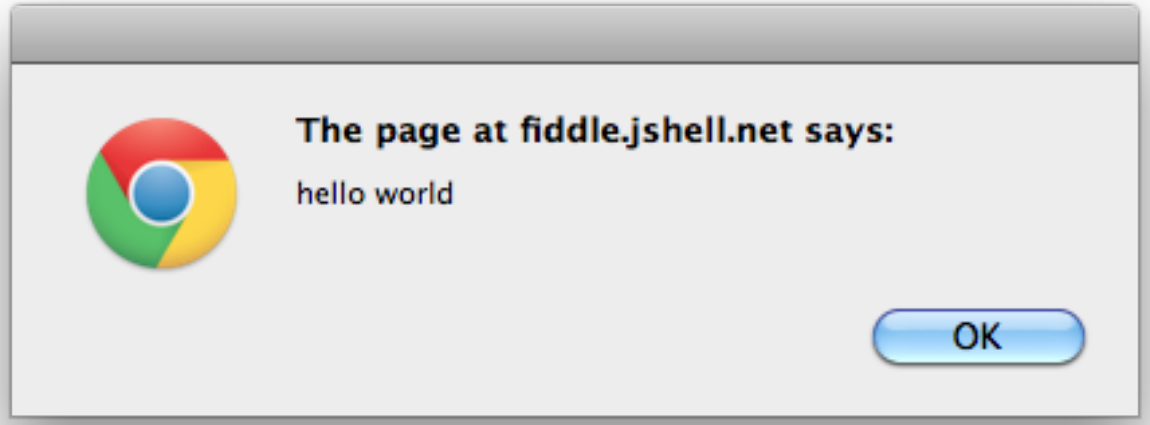
Alerts

Javascript can create alert boxes.

This command:

```
alert("hello world");
```

results in:



Variables

To create a variable, you use the keyword **var**.

```
var myNum = 6;
```

And very importantly, don't forget the **semi-colon**!

Every **stand-alone command** must have a **semi-colon**. This means **`ifs`** and **`fors`** don't need them, but the commands inside them do!

Ifs

Just like in Scratch, we can use **ifs**.

```
if ( 6 > 7 )
{
    alert("Six is bigger than seven!?!?!");
}
else
{
    alert("Seven is bigger than six!");
}
```

This will create a box of "Seven is bigger than six!"

All together now!

These can all be combined together! Here is what the previous page's code might look like if we included variables as well!

```
var myNum = 6;
var largerNum = 7;
if ( myNum > largerNum )
{
    alert("Six is bigger than seven!?!?!?");
}
else
{
    alert("Seven is bigger than six!");
}
```

Interacting with HTML

Javascript is commonly used on webpages. Something useful it can do is change the contents of HTML elements.

The javascript code:

```
document.getElementById("toChange").innerHTML = 'Hello there!';
```

finds the HTML element with the **id**, "toChange", and changes its HTML.

Javascript Worksheet

With your partner, work through the Introduction to Javascript Worksheet.

The Form Block

To create an interactive form in HTML, we use the <form> block:

```
<form>
    ...
    Add user input here
    ...
</form>
```

Input Tag

Inside **<form></form>** block, we use many types of **<input>** tags that make our form interactive in different ways.

Input tags each have a **type** and a **name** and they look like this:

```
<input type="text" name="firstName">
```

Input blocks do not require a closing tag!

Getting User Input

For each **type** of form we learned, we can find out the user's input into the form. This is called the **value** of the **<input>** object.

For example, if we created a "text" input box with the **id** "firstName," we could find out what the user typed in the box by creating a variable in our JavaScript code as such:

```
var firstName = document.getElementById("firstName").value;
```

On-Click Events

For "button" **<input>** objects, we can make events happen when we click them by adding an **onClick** function into their tags. In this example, we named the function doSomething().

```
<form>

<input  id="firstName" type="text">

<input type="button" value="Click Me!" onClick="doSomething()">

</form>
```

Defining Our Function

Once we declare an **onClick** function, we need to define what the function does in our JavaScript code. Let's make doSomething() create an alert that prints the variable firstName that we created earlier:

```
function doSomething() {

    var firstName = document.getElementById("firstName").
value;

    alert( firstName );

}
```

Improving Your Recipe with Javascript

We've already covered HTML and CSS with your recipe - now let's add Javascript!

Follow along with the worksheet to let users input how many servings they want and update the ingredients accordingly.

Tomorrow's tours...

- Early morning tour here
- Leave here @ 9:30 am
- 10:30-11am arrival

The Group: 25 people

Agenda: Friday, July 13, 2012
10:30am - Arrive: Karen will have parking passes f
how many you'll require
11:00am - Tech Talk
11:45am - Google Panel Discussion
12:30pm - Lunch
1:30pm - Tour
2:30pm - Depart

Library Functions

The makers of JavaScript have provided us with lots of pre-written useful functions to use in our own code!

These functions include ways to handle strings, such as finding their length, as well as simple math operations.

Finding the Length of a String

A string is a type of object in JavaScript.

Strings can contain letters, numbers, or characters and are written in quotes. For example:

"This is a string" and "These" "are" "strings!"

For any string, we can find its length by writing **.length** after it.

"hello".length will return 5.
"I am a string".length will return 13. (Spaces count!)

Finding Characters in Strings

We can also pull out the character at any **index** in a string using the function **charAt**. The **index** of a character is the number of its place in a string.

In JavaScript and many other languages, we start counting at zero, so the 1st character in a string has index 0, the second has index 1, and so on.

To find the 1st character of "hello", we would call
"hello".charAt(0). This would return 'h'.

Strings as Variables

We can store strings as variables like this:

var myWord = "hello";

Then, we can call our string functions on the variable name **myWord** instead of on "hello".

myWord.length returns 5.
myWord.charAt(4) returns 'o'.

This can be useful for very long strings!

Absolute Value of Numbers

The JavaScript **Math** library contains many functions for math operations.

To find the **absolute value** of a number, we write

Math.abs(number)

Math.abs(-10) returns 10.
Math.abs(4) returns 4.

Powers of Numbers

We can evaluate exponents in JavaScript using the **Math** function **Math.pow(a,b)**.

The first number is the base, while the second is the power.

Math.pow(3,2) returns 3 to the 2nd power, or 9.

Math.pow(5,1) returns 5 to the 1st power, or 5.

Password Analyzer

For our last Javascript activity, we'll create a way to analyze passwords from user input.

What Is An Internet Footprint?

An internet footprint is anything associated with your name on the internet.

Ex. Anything that shows up when you do a google search of your name.

Who Might Look At Your Footprint?

Recall what showed up when you google searched yourself earlier - pull it up to refresh your memory!

Would you be comfortable with these people seeing that information:

- your parents
- your friends
- your employer
- a potential employer