JavaScript Chapter 3

Andrea
Ricciardi

- ## Timers

  Consentono di attivare qualsiasi tipo di codice JavaScript, dopo un determinato periodo di tempo.

- ## Browser Metrics

  Le grandezze di un'immagine di una pagina web, possono essere stabilite a priori in base alla dimensione della finestra del browser, oppure dinamicamente adattando il contenuto in base al suo ridimensionamento.

- ## Cookies & Browser History

  I *cookies* sono come delle variabili che vengono memorizzate sul disco rigido dell'utente dal browser. Durano oltre una singola sessione web. Sono usati per la tracciatura di sessioni e memorizzazione di informazioni riguardanti gli utenti che accedono ad un server.

Client Server & JavaScript

- Quando si clicca su un link o si digita un URL, il browser richiede una pagina da un server.
- Una volta che la pagina è stata consegnata al browser, il server è in gran parte escluso da ogni compito. Il codice infatti, viene eseguito direttamente sul client.
- Il codice JavaScript nella pagina lavora in contemporanea con il browser web per rispondere all'interazione dell'utente e se necessario modificare la pagina.
- La parte del browser web che esegue codice JavaScript è chiamata **JavaScript Interpreter.**

Client , Server & JavaScript

Request Page

```
GET / HTTP/1.1
Hoot: www.duncunsdonuts.com
Connection: close
Accept-Encoding: gzip
Accept: image/gif, image/x-xbitmap, image/jpeg .....
Accept-Language: en-us
User-Agent : Mozilla/5.0 (Macintosh; U; Intel Mac OS X; en-US; rv:1.8.1.7)...
```

Il browser richiede la pagina dal server

Serve Page

```
<html>
    <head>
        <title>Duncan's Just-In-Time Donuts</title>
        <!-- CSS-->
        <link rel="stylesheet" type="text/css" href="donuts.css">
        <script type ="text/javascript">
            function updateOrder(){
            .....
            }
            function parseDonuts(donutString){
            .....
            }
            function placeOrder(){
            .....
            }
        </script>
    </head>
    <body>
        <div id="frame">
        ...
        ....
    </body>
</html>
```

Il server risponde con la pagina richiesta

Timer

- Con l'utilizzo dei *timer* è possibile eseguire qualsiasi tipo di codice JavaScript a intervalli regolari o dopo un tempo prestabilito.

- I due passi principali per settare i timer sono:
  - stabilire un «time delay»;
  - permettere al timer di sapere quale parte di codice deve essere eseguita una volta trascorso il tempo.

- Il conto alla rovescia inizia dal momento in cui viene settato il timer.

***tempo espresso in millisecondi (1s x 1000 = 1ms)**

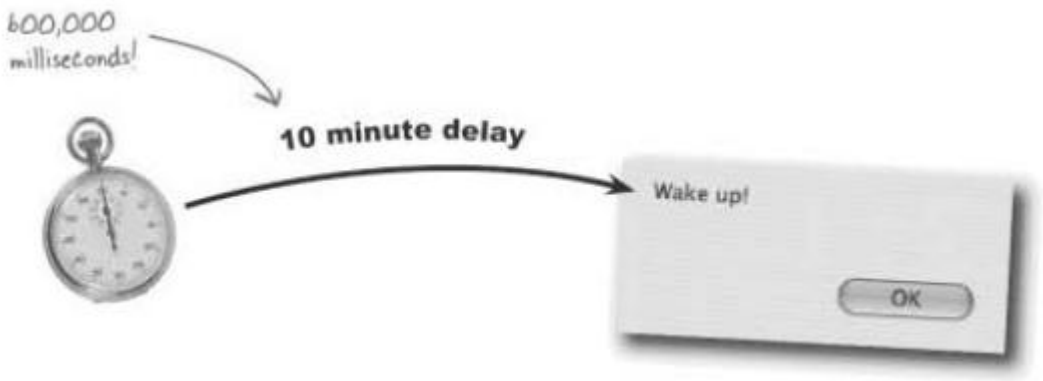
Timer - setTimeout();

- *setTimeout();* permette di eseguire del codice dopo un determinato lasso di tempo.

```
setTimeout("Timer code", Timer delay);
```

Es.

```
setTimeout("alert('Wake up!');", 10 * 60 * 1000);
```



TIMER – SETINTERVAL();

- *setInterval();* permette l'impostazione di un timer a intervalli, in questo modo il codice verrà eseguito ripetutamente.

Es.

```
var timerID = setInterval("alert('Wake up!');", 600000);
```

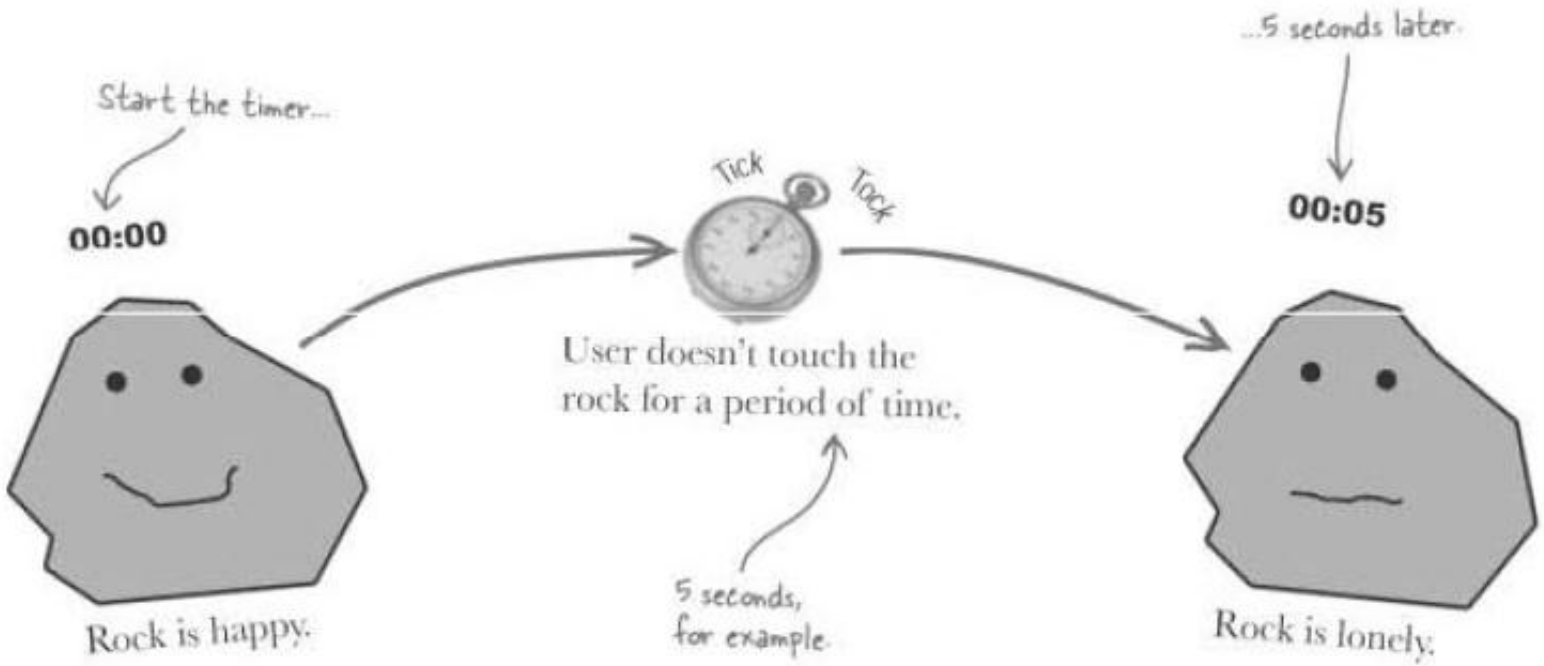
***in questo caso salviamo il timer ID in una variabile.**

Timer

```
<html>
    <script type="text/javascript">window["_gaUserPrefs"] = { ioo : function() { return true; } }</script>
<head>
    <title>iRock - The Virtual Pet Rock</title>
    <script type="text/javascript">
      var userName;

      function greetUser() {
        alert('Hello, I am your pet rock.');
      }

      function touchRock() {
        if (userName) {
          alert("I like the attention, " + userName + ". Thank you.");
        }
        else {
          userName = prompt("What is your name?", "Enter your name here.");
          if (userName)
            alert("It is good to meet you, " + userName + ".");
        }
        document.getElementById("rockImg").src = "rock_happy.png";
        setTimeout("document.getElementById('rockImg').src = 'rock.png';", 5 * 60 * 1000);
      }
    </script>
  <style type="text/css"></style>
 </head>

  <body onload="greetUser();">
    <div style="margin-top:100px; text-align:center">
      <img id="rockImg" src="rock_happy.png" alt="iRock" style="cursor:pointer" onclick="touchRock();">
    </div>
</body>
</html>
```

iRock & Timer

- http://www.headfirstlabs.com/books/hfjs/ch03/iroc



Browser Metrics

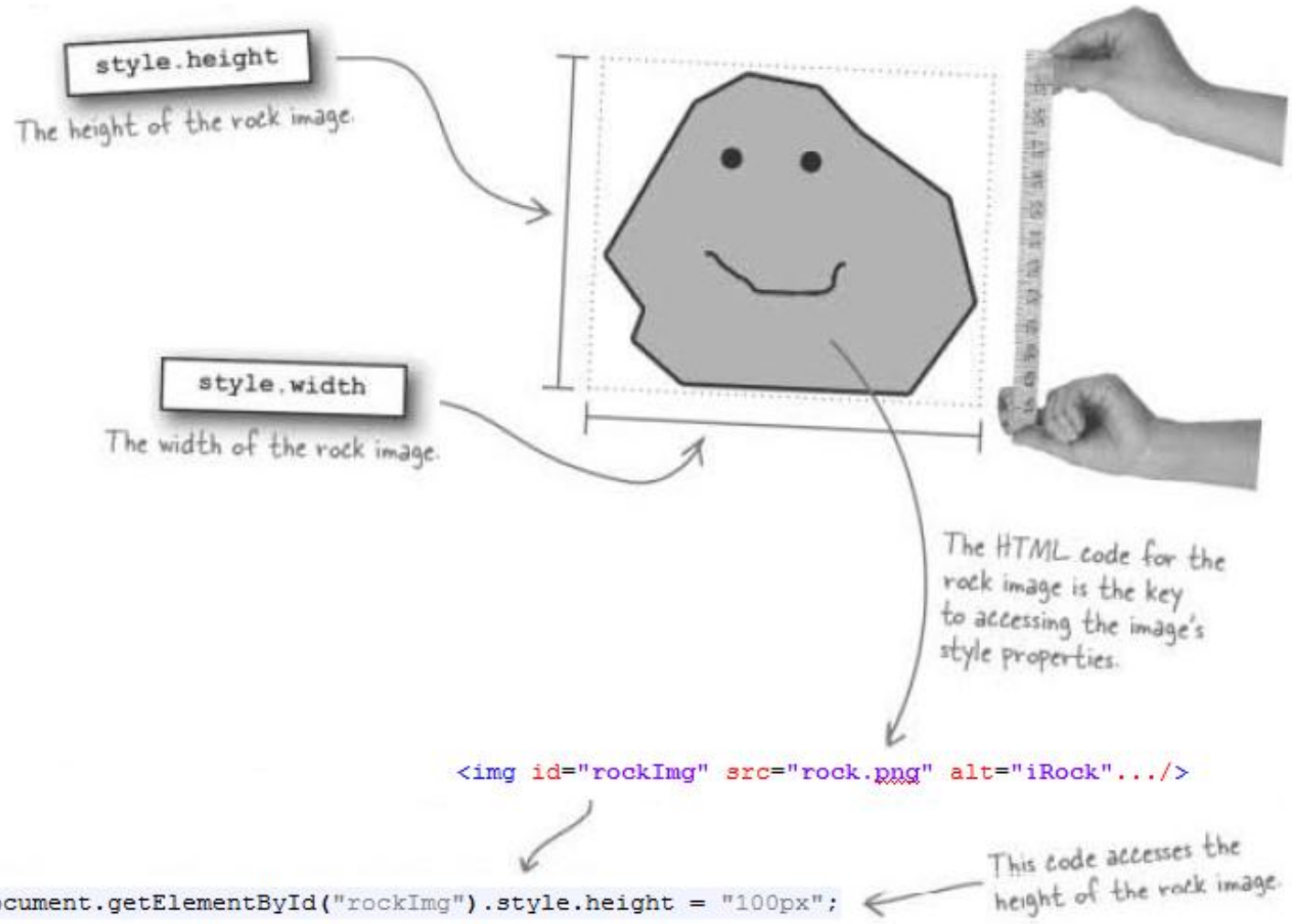
- JavaScrpt permette che il contenuto della nostra pagina web si adatti perfettamente alle dimensioni della finestra del browser.

- Tramite l'oggetto *document si* accedere alla finestra client.

- Le proprietà *body.clientWidth* e *body.clientHeight* del documento permettono di manipolare la larghezza e l'altezza della finestra.

Browser Metrics

- Si possono determinare le dimensioni di un elemento in una pagina web, sia in base alla grandezza della pagina sia dinamicamente.



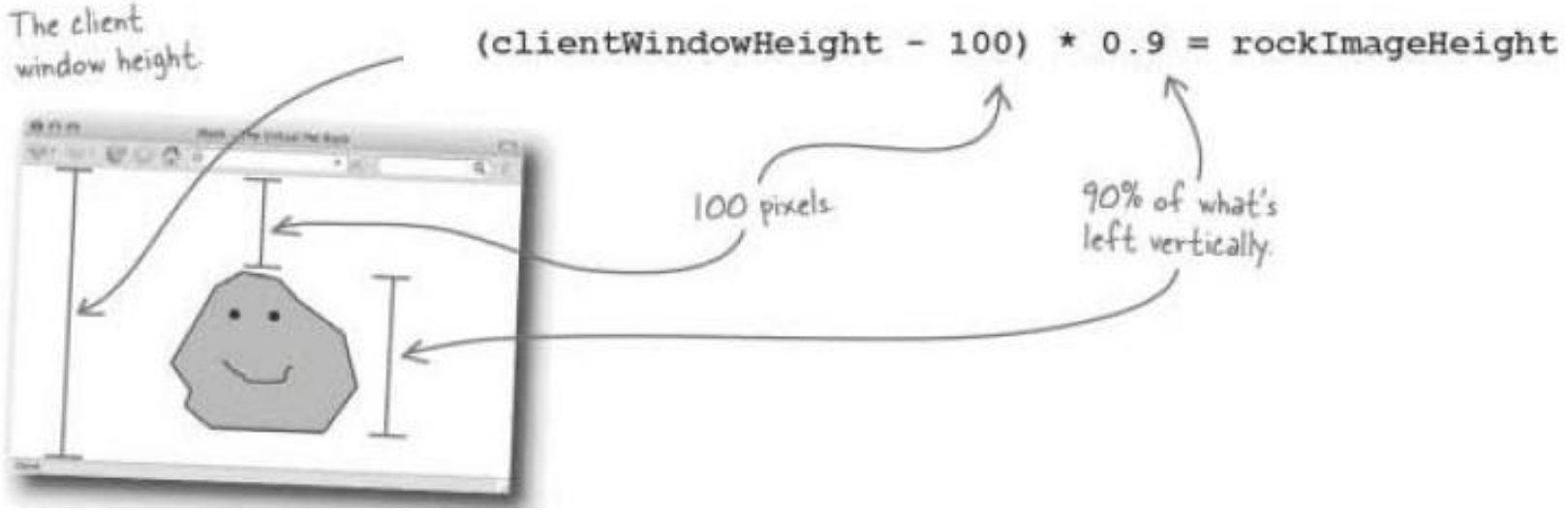
Browser Metrics

- È possibile dare uno *stile* ad ogni elemento di una pagina web, in modo da poter accedere alla larghezza e all'altezza di ognuno di essi.

- Per settare la grandezza dell'oggetto non è necessario specificare entrambe le dimensioni, poiché altre proprietà provvederanno in automatico a settare le giuste proporzioni in base all'unica grandezza specificata.

Browser Metrics

- Per fare in modo che l'immagine si adatti al ridimensionamento di una finestra, bisogna settare dei parametri specifici.



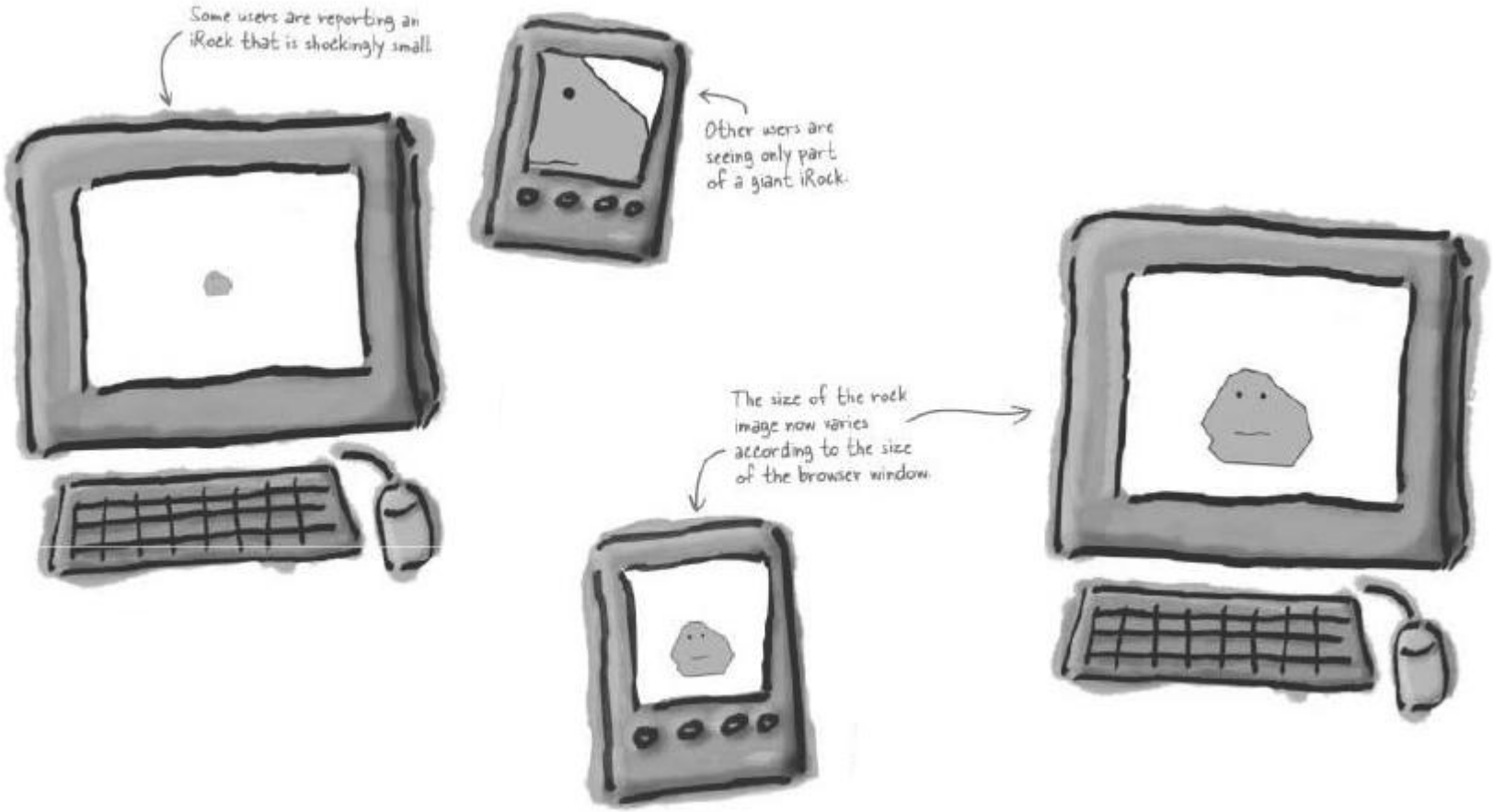
- Molto importante è anche la gestione degli eventi tramite *onload e onresize*.

Browser Metrics

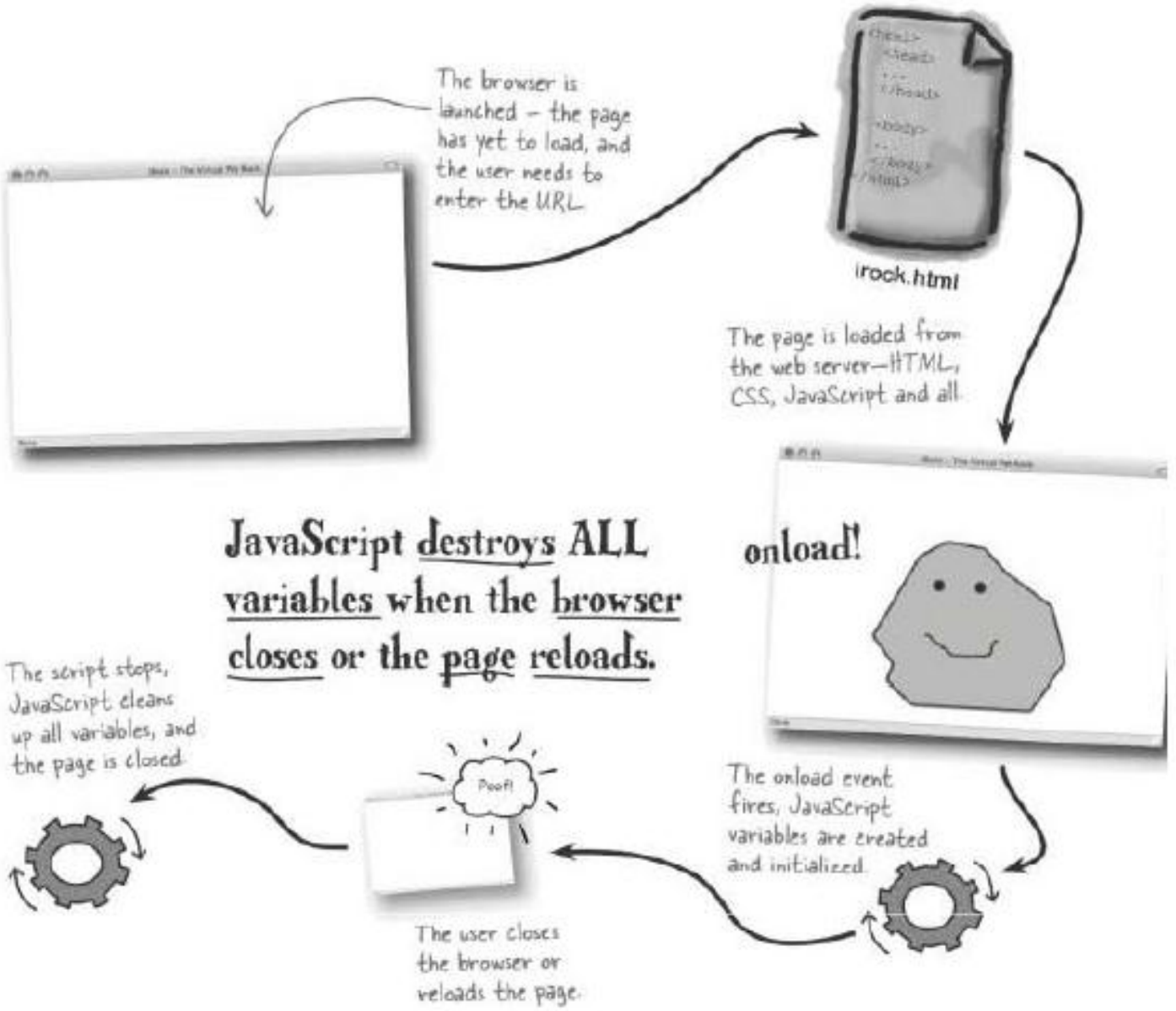
```
<html>
<script type="text/javascript">window["_gaUserPrefs"] = { ioo : function() { return true; } }</script><head>
    <title>iRock - The Virtual Pet Rock</title>
    <script type="text/javascript">
      var userName;

      function resizeRock() {
        document.getElementById("rockImg").style.height = (document.body.clientHeight - 100) * 0.9;
      }

      function greetUser() {
        alert('Hello, I am your pet rock.');
      }

      function touchRock() {
        if (userName) {
          alert("I like the attention, " + userName + ". Thank you.");
        }
        else {
          userName = prompt("What is your name?", "Enter your name here.");
          if (userName)
            alert("It is good to meet you, " + userName + ".");
        }
        document.getElementById("rockImg").src = "rock_happy.png";
        setTimeout("document.getElementById('rockImg').src = 'rock.png';", 5 * 60 * 1000);
      }
    </script>
  <style type="text/css"></style></head>

  <body onload="resizeRock(); greetUser();" onresize="resizeRock();">
    <div style="margin-top:100px; text-align:center">
      <img id="rockImg" src="rock_happy.png" alt="iRock" style="cursor: pointer; height: 488.7px;" onclick="touchRock();">
    </div>
</body>
</html>
```

iRock & Browser Metrics

- http://www.headfirstlabs.com/books/hfjs/ch03/irock/irock6.html



Cookies & Browser History



Cookies & Browser History

- I cookies offrono un modo per memorizzare i dati in modo "persistente", oltre il singolo ciclo di vita di uno script.

- Sono dati memorizzati dal browser nel computer dell'utente, per rendere disponibili le informazioni inserite da quest'ultimo, prima della chiusura della pagina, di un "reload" o dell'arresto del sistema.

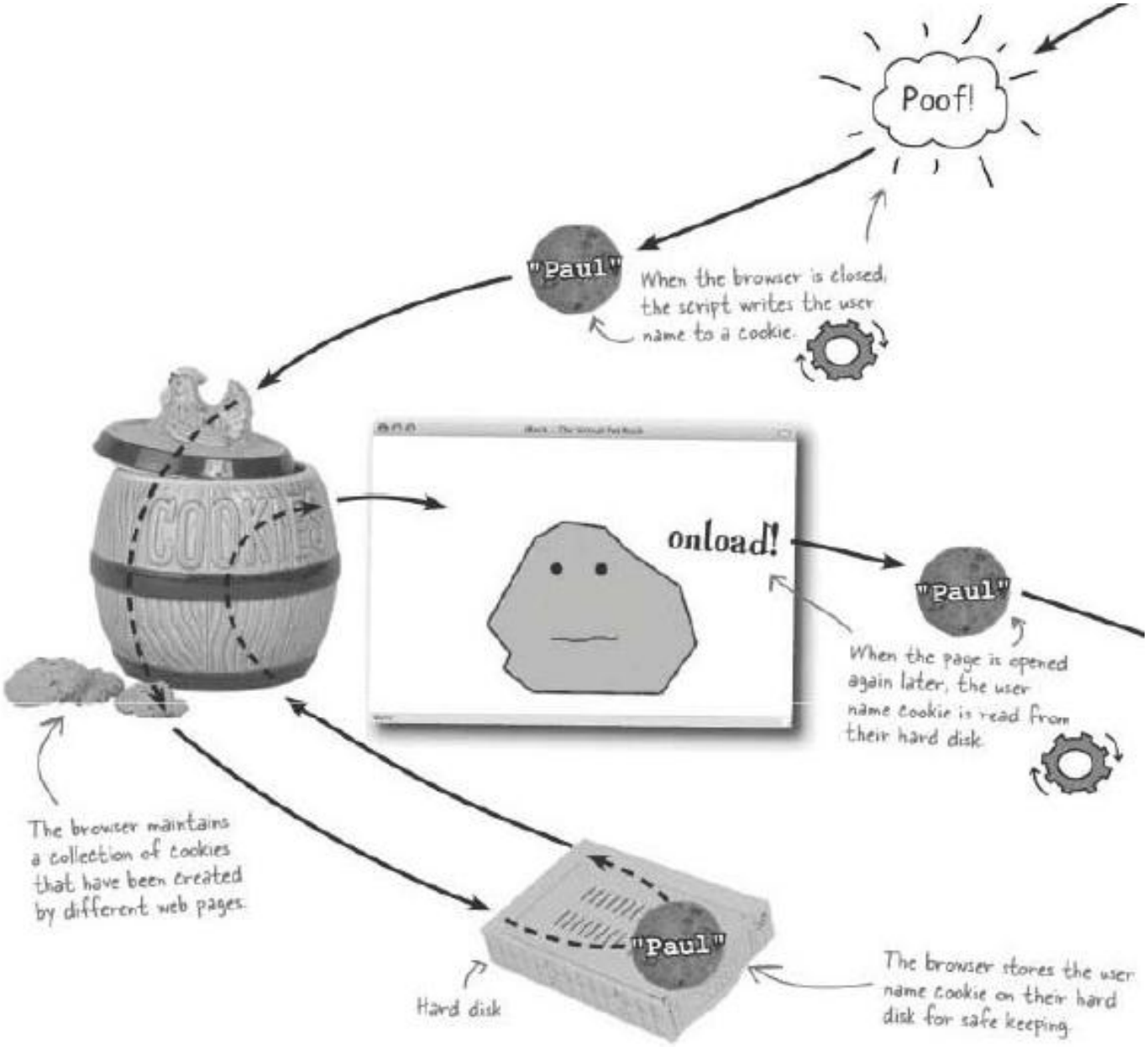
Cookies & Browser History

- I cookies salvano singole informazioni attraverso nomi unici, proprio come le variabili, ma a differenza di quest'ultime hanno una data di scadenza.

- Quando un cookie scade, viene distrutto e l'informazione salvata viene persa.

- I cookies sono memorizzati nel HD dell'utente come una grande stringa di testo associata al sito web o al dominio. Ognuno è identificato da una *chiave* per permetterne il riconoscimento e sono divisi tra loro da un ' ; '.

Cookies & Browser History



Write Cookie

```
function writeCookie(name,value,days){
    //Di default, non c'è una data di scadenza quindi il cookie è temporaneo
    var expires = "";

    /*Specifichiamo il numero dei giorni
    durante il quale i cookie risulta persistente*/

    if(days){
        var date = new Date();
        date.setTime(date.getTime() + (days *24*60*60*1000));
        expires = "; expires="+ date.toGMTString();
    }
    //Settiamo il cookie con nome, valore e data di scadenza
    document.cookie = name + "=" + value + expires + "; path=/";
}
```

- La data di scadenza è calcolata convertendo il numero di giorni in ms, con l'aggiunta dell'ora corrente.

Read Cookies

```
function readCookie(name){
    //Ricerca di uno specifico cookie e restituzione del suo valore
    var searchNme = name + "=";
    var cookies = document.cookie.split(';');
    for(var i=0; i<cookies.lenght; i++){
        var c = cookies[i];
        while(c.charAT(0) == ' ')
            c = c.substring(1,c.lenght);
        if (c.indexOF(serchName) == 0)
            return c.substring(searchName.lenght, c.lenght);
    }
    return null;
}
```

- Tramite *.split(';');* ogni cookie è diviso da un ' ; '

Erase Cookies

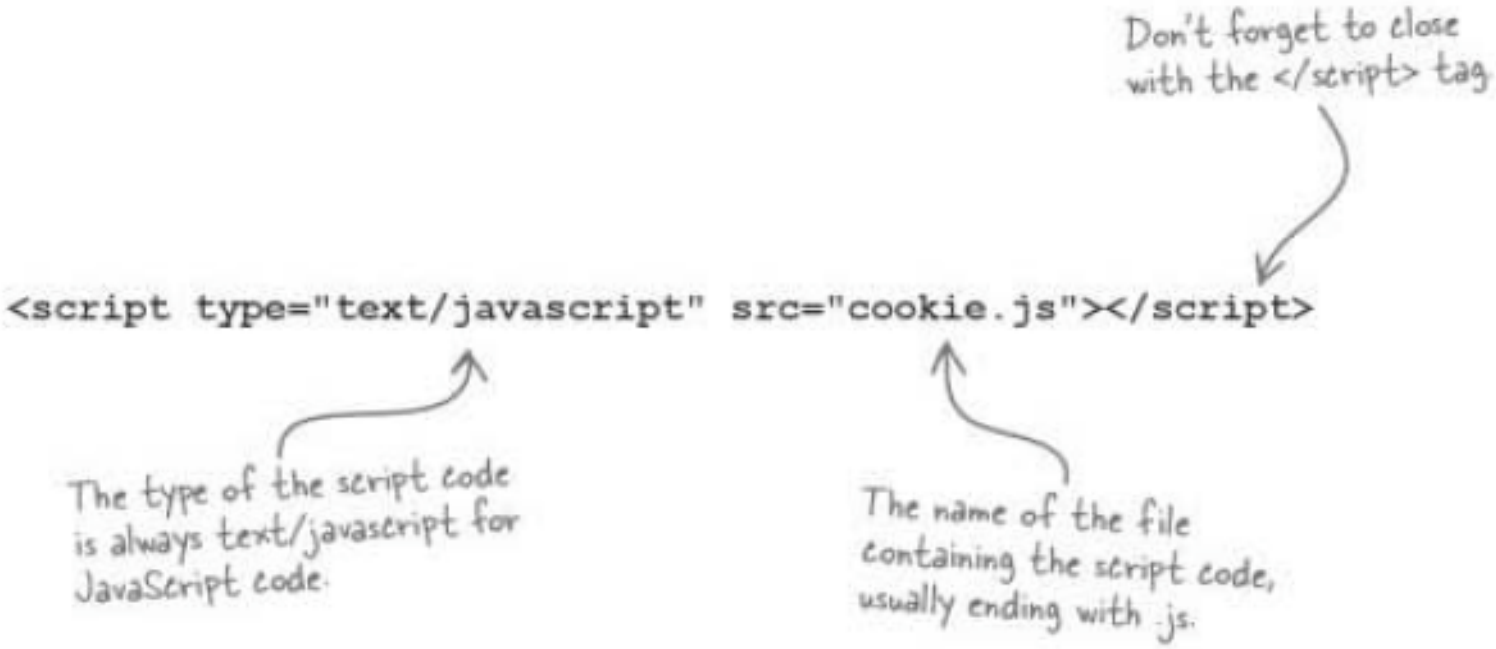
```
function eraseCookie(name){
    //Cancella un cookie specifico
    writeCookie (name, "", -1);
}
```

- È possibile cancellare un cookie riscrivendolo senza valore e con una data di scadenza pari a -1.

Import Cookies

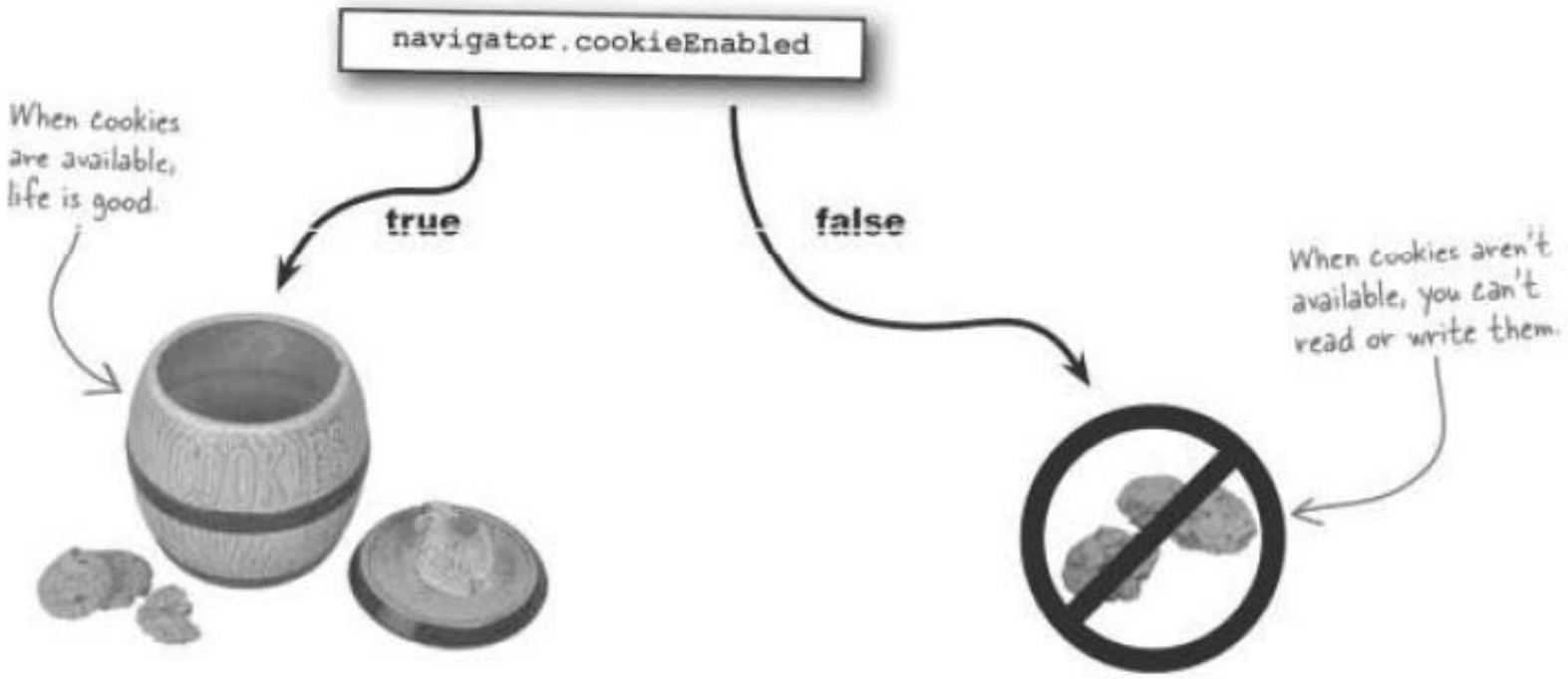
- Una volta scritto il codice JavaScript relativo ai cookies e salvato in un *file.js*, c'è bisogno di importarlo nella pagina html desiderata.

Es. cookie.js



Cookie Enabled

- Nel caso in cui i cookies siano disabilitati, il nostro programma non funzionerà più correttamente, quindi ricorriamo ad una proprietà booleana del browser.



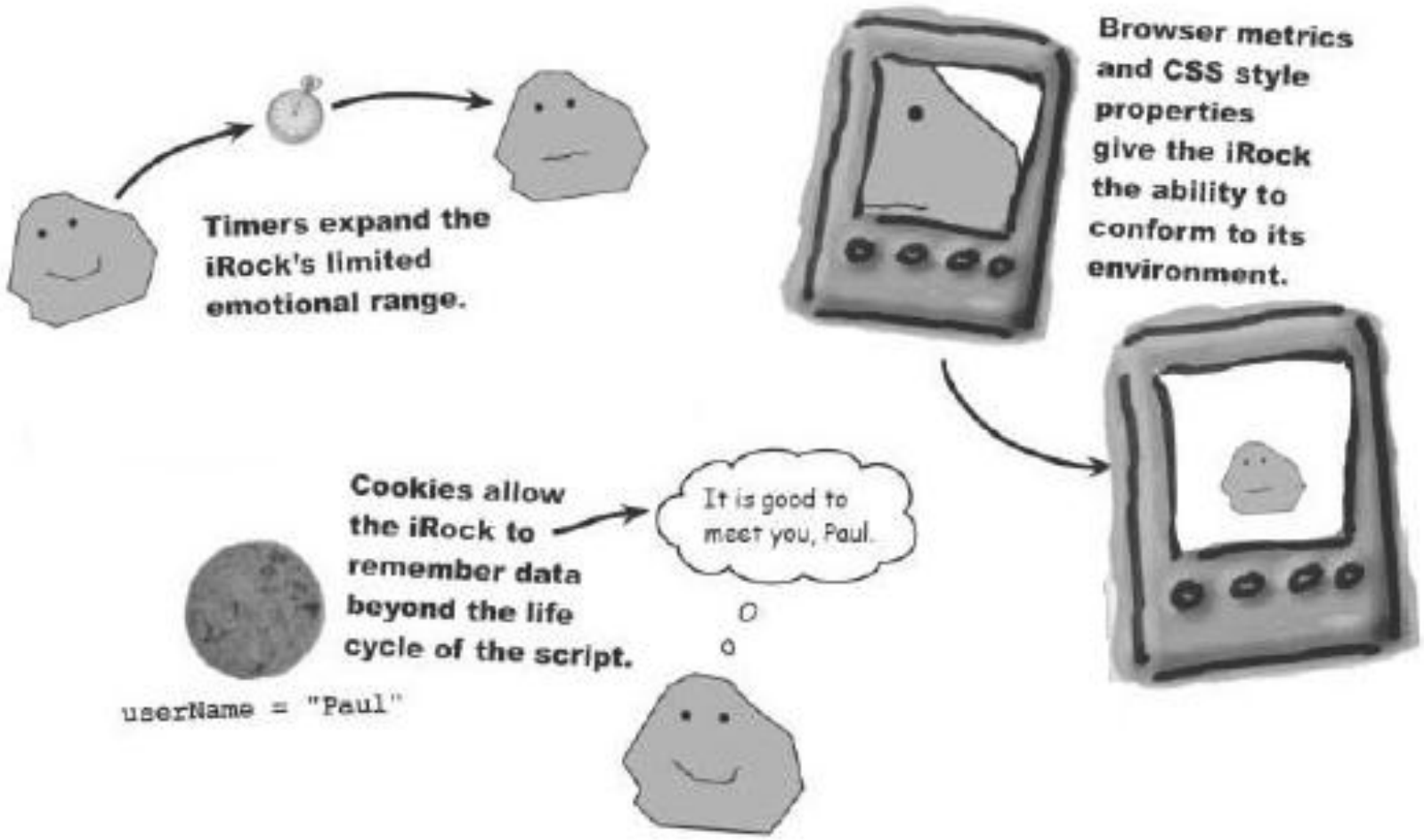
Cookie Enabled

```
...
function greetUser() {
  if (navigator.cookieEnabled)
    userName = readCookie("irock_username");
  if (userName)
    alert("Hello " + userName + ", I missed you.");
  else
    alert('Hello, I am your pet rock.');
}

function touchRock() {
  if (userName) {
    alert("I like the attention, " + userName + ". Thank you.");
  }
  else {
    userName = prompt("What is your name?", "Enter your name here.");
    if (userName) {
      alert("It is good to meet you, " + userName + ".");
      if (navigator.cookieEnabled)
        writeCookie("irock_username", userName, 5 * 365);
      else
        alert("Cookies aren't supported/enabled in your browser, which means I won't remember you later. I'm sorry.");
    }
  }
  document.getElementById("rockImg").src = "rock_happy.png";
  setTimeout("document.getElementById('rockImg').src = 'rock.png';", 5 * 60 * 1000);
}
...
```

http://www.headfirstlabs.com/books/hfjs/ch03/irock/irock8.html

Cookies affect browser security

- I cookies sono solamente una stringa di testo salvata in un file del proprio hard disk e non possono essere manipolati da nessun altro.
- Dal momento che non sono programmi eseguibili, non possono diffondere ne virus ne worms.
- Possono memorizzare dati personali, ma solo quando gli utenti li hanno consapevolmente inseriti in una pagina web.
- In tutti i modi i cookies non sono considerati strumenti sicuri per la memorizzazione e non è affatto una buona idea salvare dati sensibili in essi.


Timers expand the iRock's limited emotional range.
Browser metrics and CSS style properties give the iRock the ability to conform to its environment.
Cookies allow the iRock to remember data beyond the life cycle of the script.
It is good to meet you, Paul.
userName = "Paul"