CRIAR COOKIE:

Crie um cookie com JavaScript

JavaScript pode criar, ler e excluir cookies com a `document.cookie` propriedade

Com JavaScript, um cookie pode ser criado assim:

```
document.cookie = "username=John Doe";
```

Você também pode adicionar uma data de validade (no horário UTC). Por padrão, o cookie é excluído quando o navegador é fechado:

```
document.cookie = "username=John Doe; expires=Thu, 18 Dec 2013 12:00:00 UTC";
```

Com um parâmetro path, você pode dizer ao navegador a que caminho o cookie pertence. Por padrão, o cookie pertence à página atual.

```
document.cookie = "username=John Doe; expires=Thu, 18 Dec 2013 12:00:00 UTC; path=/";
```

LER COOKIE:

Leia um cookie com JavaScript

Com JavaScript, os cookies podem ser lidos assim:

```
var x = document.cookie;
```

`document.cookie` retornará todos os cookies em uma sequência semelhante a: cookie1 = value; cookie2 = valor; cookie3 = valor;

ALTERAR COOKIE:

Alterar um cookie com JavaScript

Com o JavaScript, você pode alterar um cookie da mesma maneira que o cria:

```
document.cookie = "username=John Smith; expires=Thu, 18 Dec 2013 12:00:00 UTC; path=/";
```

O cookie antigo é substituído.

EXCLUIR COOKIE:

Você não precisa especificar um valor de cookie ao excluir um cookie.

Basta definir o parâmetro expires para uma data passada:

```
document.cookie = "username=; expires=Thu, 01 Jan 1970 00:00:00 UTC; path=/;";
```

Você deve definir o caminho do cookie para garantir que você exclua o cookie correto.

Alguns navegadores não permitem excluir um cookie se você não especificar o caminho.

exemplo: criando cookie com javascript:

No exemplo a seguir, criaremos um cookie que armazena o **nome de um visitante.**

Na primeira vez em que um visitante chegar à página da Web, será solicitado que ele preencha seu nome. O nome é então armazenado em um cookie.

Na próxima vez que o visitante chegar à mesma página, receberá uma mensagem de boas-vindas.

Para o exemplo, criaremos três funções JavaScript:

1. Uma função para definir um valor de cookie
2. Uma função para obter um valor de cookie
3. Uma função para verificar um valor de cookie

1)

Uma função para definir um cookie

Primeiro, criamos um functionque armazena o nome do visitante em uma variável de cookie:

Exemplo

```
function setCookie(cname, cvalue, exdays) {
  var d = new Date();
  d.setTime(d.getTime() + (exdays*24*60*60*1000));
  var expires = "expires="+ d.toUTCString();
  document.cookie = cname + "=" + cvalue + ";" + expires + ";path=/";
}
```

Exemplo explicado:

Os parâmetros da função acima são o nome do cookie (cname), o valor do cookie (cvalue) e o número de dias até que o cookie expire (exdays).

A função define um cookie, adicionando o nome do cozinheiro, o valor do cookie e a cadeia de caracteres expirada.

2)

Uma função para obter um cookie

Em seguida, criamos um function que retorna o valor de um cookie especificado:

Exemplo

```
function getCookie(cname) {
  var name = cname + "=";
  var decodedCookie = decodeURIComponent(document.cookie);
  var ca = decodedCookie.split(';');
  for(var i = 0; i <ca.length; i++) {
    var c = ca[i];
    while (c.charAt(0) == ' ') {
      c = c.substring(1);
    }
    if (c.indexOf(name) == 0) {
      return c.substring(name.length, c.length);
    }
  }
```

```
  return "";
}
```

Função explicada:

Tome o nome da cook como parâmetro (cname).

Crie uma variável (nome) com o texto a ser pesquisado (cname + "=").

Decodifique a sequência de cookies, para manipular cookies com caracteres especiais, por exemplo, '$'

Divida document.cookie em ponto e vírgula em uma matriz chamada ca (ca = decodedCookie.split (';')).

Faça um loop na matriz ca (i = 0; i <comprimento ca; i ++) e leia cada valor c = ca [i]).

Se o cookie for encontrado (c.indexOf (name) == 0), retorne o valor do cookie (c.substring (name.length, c.length).

Se o cookie não for encontrado, retorne "".

3)

Uma função para verificar um cookie

Por fim, criamos a função que verifica se um cookie está definido.

Se o cookie estiver definido, ele exibirá uma saudação.

Se o cookie não estiver definido, ele exibirá uma caixa de prompt, solicitando o nome do usuário, e armazenará o cookie de nome de usuário por 365 dias, chamando a `setCookie`função:

Exemplo

```
function checkCookie() {
  var username = getCookie("username");
  if (username != "") {
   alert("Welcome again " + username);
  } else {
    username = prompt("Please enter your name:", "");
    if (username != "" && username != null) {
      setCookie("username", username, 365);
    }
  }
}
```

TUDO JUNTO:

```
function setCookie(cname, cvalue, exdays) {
  var d = new Date();
  d.setTime(d.getTime() + (exdays * 24 * 60 * 60 * 1000));
  var expires = "expires="+d.toUTCString();
  document.cookie = cname + "=" + cvalue + ";" + expires + ";path=/";
```

```
}

function getCookie(cname) {
  var name = cname + "=";
  var ca = document.cookie.split(';');
  for(var i = 0; i < ca.length; i++) {
    var c = ca[i];
    while (c.charAt(0) == ' ') {
      c = c.substring(1);
    }
    if (c.indexOf(name) == 0) {
      return c.substring(name.length, c.length);
    }
  }
  return "";
}

function checkCookie() {
  var user = getCookie("username");
  if (user != "") {
    alert("Welcome again " + user);
  } else {
    user = prompt("Please enter your name:", "");
    if (user != "" && user != null) {
      setCookie("username", user, 365);
    }
  }
}
```

Cookies são dados salvos em pequenos arquivos de texto no computador da pessoa que utiliza *Internet*.

`allCookies = document.cookie;`

- `allCookies` é uma string contendo uma lista separada por vírgula de "cookies" (isto é, *`chave = valor`* pares).

`updatedCookie = document.cookie;`

- `updatedCookie é uma string de forma` *`chave = valor`*. Observe que você só pode definir / atualizar um cookie de cada vez usando esse método.

- Qualquer um dos seguintes valores de atributo cookie pode, opcionalmente, seguir o valor-chave par, especificando o cookie para definir / atualizar, e precedido por um ponto e vírgula :
    - `;path = caminho` (Por exemplo, '/' , '/meuDiretorio'). Se não for especificado, o padrão é o caminho atual do local do documento atual.
    - `;domain = `*`domínio`* (por ex, 'exemplo1.com ', '.exemplo1.com', (inclui todos os subdomínios), 'subdominio.exemplo1.com'). Se não for especificado, o padrão é a parte do host local do documento atual.
    - `;max-age = maxima-idade-`*`em-segundos`* (Por exemplo, 60 * 60 * 24 * 365 para um ano)
    - `;expires = `*`data-em-formato-GMTString`* (Poderia usar Date.toGMTString, agora obsoleto). Se não for especificado ele expira no final da sessão.
    - `;secure` (cookie só podem ser transmitidos através do protocolo seguro como https)
- A cadeia de valor do cookie pode usar encodeURIComponent() para garantir que a cadeia não contenha nenhuma vírgula, ponto-e-vírgula, ou espaços em branco (que não são permitidos nos valores de cookie).

```
1. document.cookie = "nome = Italo";
2. document.cookie = "comida_favorita = lasanha";
3. alert(document.cookie);
4. // Mostra: nome = Italo; comida_favorita = lasanha
```

É importante notar que o path **não** protege contra a leitura não autorizada do cookie de um caminho diferente**.** Ele pode ser facilmente contornado com DOM simples (por exemplo, a criação de um elemento iframe oculto com o caminho do cookie, e depois aceder a este iframe `contentDocument.cookie`propriedade). Ele pode ser facilmente (por exemplo, a criação de um elemento o caminho do cookie, e depois aceder a este A única maneira de proteger o acesso "cookie" é usando um domínio ou subdomínio diferente , devido à política de mesma origem.