Что такое JavaScript

JavaScript — это наиболее популярный, относительно простой *объектно-ориентированный язык*, предназначенный для создания *небольших клиентских* и серверных приложений для Internet.

Разработан фирмой Netscape в сотрудничестве с Sun Microsystems и анансирован в 1995 году.

Работает в большинстве браузеров (Intenet Explorer, Firefox, Chrome, Opera, и Safari).

Что Вы Уже Должны Знать

базовое представление в:

- HTML
- CSS

Особенности JavaScript

Программы, написанные на языке JavaScript (называются «скриптами»), *включаются в состав HTML-документов* и распространяются вместе с ними.

Браузеры распознают встроенные в текст документа script-коды и выполняют их, как только загружается страница.

Таким образом, JavaScript — *интерпретируемый язык* программирования (скрипты исполняются без предварительной компиляции).

Для создания программ на JavaScript *не требуется никаких дополнительных средств* — необходим лишь *браузер (*с активированной способностью распознавать JavaScript*)* и текстовый редактор, позволяющий создавать HTML-документы.

Язык JavaScript **регистрозависимый**, т.е. заглавные и прописные буквы алфавита считаются разными символами.

Отличие от C и Java

В JavaScript, в отличие от C и Java, *понятие классов не является основой синтаксических конструкций* языка. Такой основой является:

- небольшой *набор предопределенных типов данных*, поддерживаемых исполняемой системой: числовые, булевские и строковые;
- *функции*, которые могут быть как самостоятельными, так и методами объектов;
- *объектная модель* с большим набором предопределенных объектов со своими свойствами и методами.

Возможности JavaScript

- ***контроль работы браузера*** (вывод диалоговых окон, открытие / закрытие окон браузера, управление режимами прокрутки и размерами окон)
- ***взаимодействие с содержимым документов*** (объект Document - чтение / изменение частей HTML-документа , объект Form - чтение / изменение содержимого его объектов Button, Checkbox, Hidden, Password, Radio, Reset, Select, Submit, Text и Textarea и т.д.)
- ***взаимодействие с пользователем*** (возможность определять обработчики событий, проверка вводимой информации, управление отсылкой содержимого форм)
- выполнение произвольных ***математических вычислений*** + средства работы со значениями ***даты и времени.***

Специальные инструменты для анализа кода и отслеживания ошибок

В браузеры встроены или устанавливаются дополнительно специальные инструменты для анализа кода и отслеживания ошибок.

В браузере Firefox

- Установите дополнение Firebug:
- Инструменты → Дополнения;
- в строке поиска введите слово firebug;
- выполните установку дополнения.

Использование Firebug для просмотра ошибок в сценариях JavaScript:

- нажмите F12, чтобы включить Firebug;
- на вкладке Консоль кликните по треугольнику и включите ее;
- еще раз щелкните по этому треугольнику и проверьте – должен быть включен показ ошибок и предупреждений JavaScript;
- перезагрузите страницу.

В браузере Opera

- Нажмите Ctrl+Shift+I, выберите Ошибки.

JavaScript. Используемые понятия

- *Объект* – совокупность свойств, методов, коллекций и событий, предоставляемых браузером в рамках объектной модели
- *Свойство* – переменная в рамках объекта, используемая для получения значения или установки нового
- *Метод –* функция, предоставляемая объектом для выполнения каких-либо действий
- *Коллекция* – упорядоченный набор свойств
- *Событие* – какое-либо действие пользователя или момент работы браузера (для реакции на события создаются обработчики событий)

JavaScript. Используемые понятия

Литералы - это простейшие данные с которыми может работать программа.

Литералы ***целого типа*** - целые числа в представлении: десятичном,

например: 15, +5, -174.

Вещественные литералы - дробные цисла.

Целая часть отделяется от дробной точкой, например: 99.15, -32.45.

Логические значения - из два: истина (true) и ложь (false).

Строковые литералы - последовательность символов, заключенная в одинарные или двойные кавычки.

Например: "ваше имя", 'ваше имя'.

SCRIPT-вставки в HTML-документе

<script language=JavaScipt>

…

</script>

Текст программы на языке JavaScript вставляется непосредственно в файл HTML-страницы. Строки между <script> и </script> содержат JavaScript и исполняются браузером.

1. Подключение в любом месте

```
<html>
<head> …..</head>
<body>……
<script language=JavaScipt>
    //вызов функций
</script>…….
</body></html>
```

```
<html>
<body>

<h1>Моя Первая Веб Страница</h1>

<script type="text/javascript">
document.write("<p>" + Date() + "</p>");
</script>

</body>
</html>
```

SCRIPT-вставки в HTML-документе

2. Вынос скриптов в заголовок HEAD

```
<html>
<head><title>Заголовок</title>
<script language=JavaScipt>
    //функции
</script>
</head>
<body>
.....
</body>
</html>
```

```
<html>
<head><title>Заголовок</title>
<script language=JavaScipt>
document.getElementById("demo").innerHTML=Date();
</script>
</head>
<body>
<p id="demo">Это параграф.</p>
</body>
</html>
```

SCRIPT-вставки в HTML-документе

3. Внешние скрипты

```
<html>
<head><title>Заголовок</title>
<script src="/my/script.js">
</script>
</head>
<body>..... </body></html>
```

Замечание: При указании атрибута src содержимое тега *<script >…</script>* игнорируется.

Решение: два (несколько) разных тега <script>:

первый с src,

второй - с командами, которые будут выполнены после выполнения внешнего файла.

```
<html>
<head>
<script type="text/javascript"
src="xxx.js"></script>
</head>
<body>
</body>
</html>
```

- Внешние файлы JavaScript обычно содержат код, который используется на нескольких различных веб страницах.
- Внешние файлы JavaScript имеют расширение .js.
- **Замечание:** Внешний скрипт не должен содержать теги <script></script>!

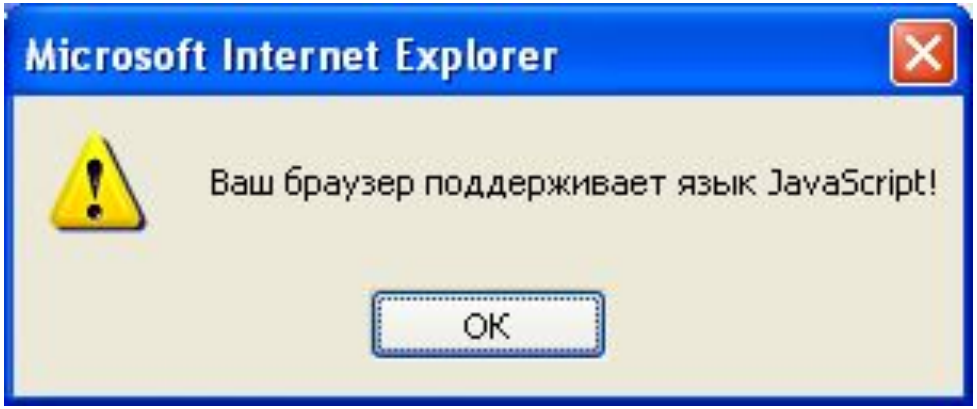
```
document.write("Это и есть JavaScript");
```

Некоторые (старые) браузеры не поддерживают JavaScript, чтобы скрыть от них вставки JavaScript, в добавляют комментарий:

```
<HTML>
  <HEAD>
……..
 <SCRIPT language=JavaScript>
    <!--
      alert("Ваш браузер поддерживает язык JavaScript!");
    //-->
    </SCRIPT>

…..

  </BODY>
</HTML>
```



Кодировка и doctype

Для гарантированного правильного отображения символов в браузере используется следующий тег, вставляемый в контейнер ***head***:

<meta charset="utf-8">

Перед тегом html указывается DOCTYPE

<!DOCTYPE HTML>

Всплывающие (диалоговые) окна

Окна оповещения - когда необходимо, чтобы пользователь обратил внимание на определенную информацию.
Когда окно оповещения будет вызвано пользователь должен будет нажать кнопку "ОК" для, того чтобы продолжить просмотр страницы.

alert("Текст окна оповещения");

```
<html>
<head>
<script type="text/javascript">
function show_alert()
{
alert("Привет! Я сигнальное окно!");
}
</script>
</head>
<body>

<input type="button" onclick="show_alert()" value="
Показать сигнальное окно" />

</body>
</html>
```

Окна подтверждения

Окна подтверждения -когда необходимо, чтобы пользователь **подтвердил или отклонил что-либо**.

Когда окно подтверждения будет вызвано пользователь должен будет нажать либо "OK", либо "Отмена", чтобы продолжить.

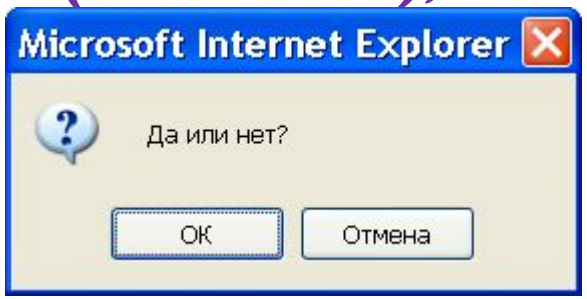
Если пользователь нажмет "OK" вернется true (истина), если пользователь нажмет "Отмена- вернется false (ложь).

var x=confirm("Текст окна подтверждения");

```
<html>
<head>
<script>
function show_confirm()
{
res=window.confirm("Да или нет?");
window.alert("res="+res);
}
</script>
</head>
<body>

<input type="button"
onclick="show_confirm()"
value="Показать окно подтверждения" />

</body>
</html>
```



Окна запроса

Окно приглашения (запроса) используется, чтобы **пользователь ввел значение** перед тем как войти на страницу.

Когда окно приглашения выскакивает, пользователь должен нажать либо "ОК" либо "Отмена" чтобы продолжить после ввода значения.

Если пользователь нажимает "**ОК**" окно возвращает введенное значение. Если же пользователь нажимает "**Отмена**" окно возвратит null (означает пустое значение, или отсутствие значения).

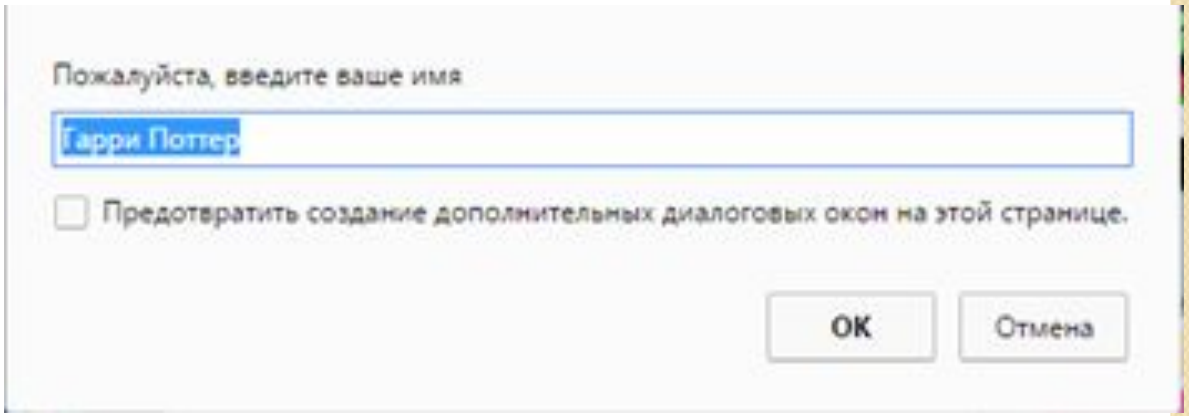
prompt(сообщение, [значение по умолчанию])

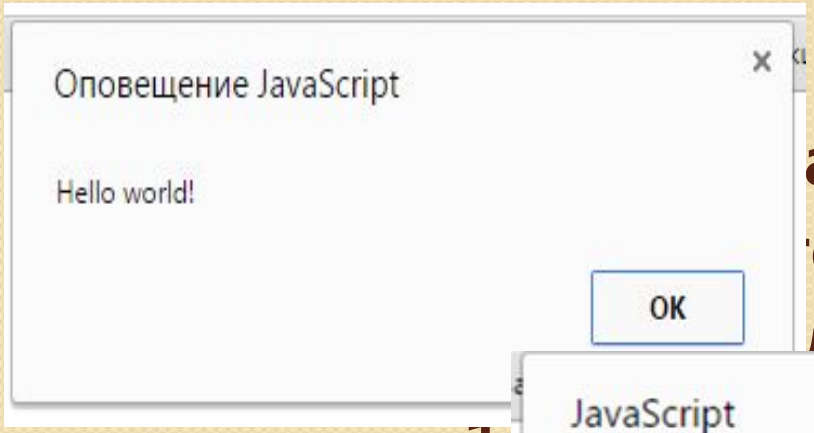
где [] - означают, что параметр необязателен, т.е. его можно опустить.

Окна запроса

```
<html>
<head>
<script type="text/javascript">
function show_prompt()
{
var name=prompt("Пожалуйста, введите ваше имя","Гарри Поттер");
document.write("Привет " + name + "! Как дела сегодня?");
}
</script>
</head>
<body>
<input type="button" onclick="show_prompt()" value="Показать окно приглашения" />

</body>
</html>
```





адание :

овые окна ввода-вывода

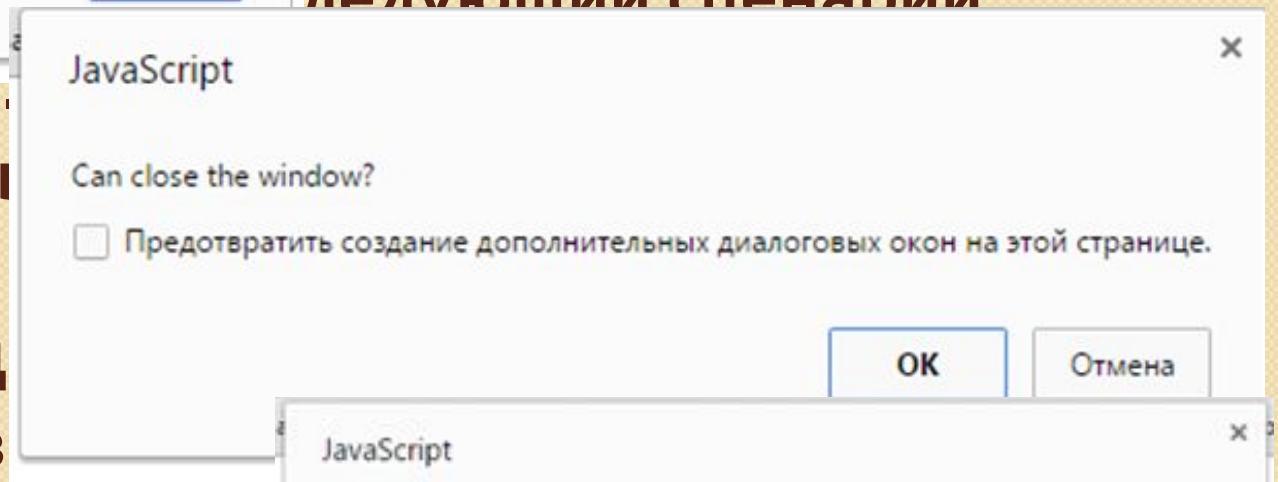
едующий сценарий:

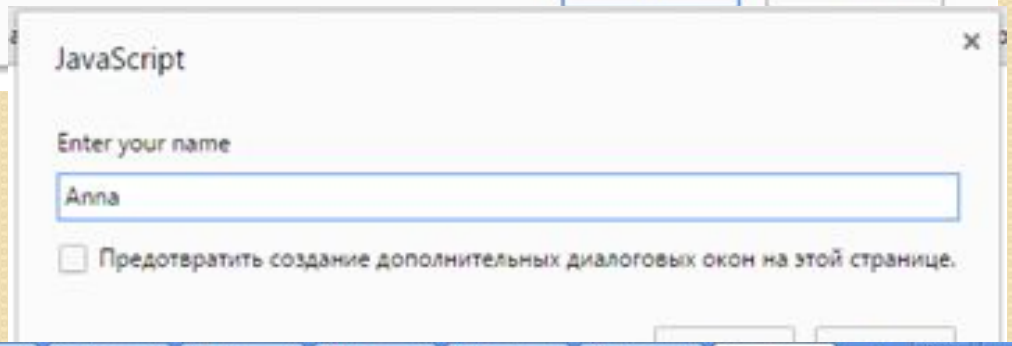
1.

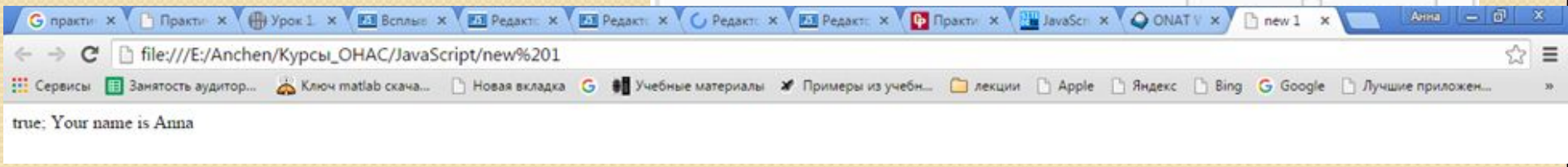
2. В

4.Вывод

в







Переменные

Переменные используются для хранения данных (значений или выражений).

Переменные определяются с помощью оператора ***var***, после которого следует имя переменной.
Переменная может иметь короткое имя, например ***x***, или более информативное имя, например ***carname*** (название автомобиля).

Правила для имен переменных JavaScript:

- Имена переменных чувствительны к регистру (y и Y это две разных переменных)
- Имена переменных должны начинаться с буквы или символа подчеркивания
- Само имя может включать буквы латинского алфавита, цифры и знак подчеркивания.

Например:

var test
var _test
var _my_test1

Переменные

После объявления переменные пусты (т.е. они пока еще не имеют значений).
Можно присвоить значения переменным, когда вы объявляете их:

```
var x=5;
var carname="Мерседес";
```

переменная **x** будет содержать значение **5**, и **carname** будет содержать значение **Мерседес**.

Оператор присвоения значения переменной обозначается символом равенства "=".
Выражение с оператором "=" интерпретатор вычисляет следующим образом: сначала вычисляется значение справа от знака равно, после чего результат присваивается переменной слева.

Замечание: Когда вы присваиваете текстовое значение переменной, заключайте его в кавычки.
Замечание: Если вы объявляете переменную повторно, она не потеряет свое значение.

Комментарии

добавляются для пояснения кода JavaScript, или чтобы сделать код более читабельным.

Однострочные комментарии начинаются с //.

```
<script type="text/javascript">
// Выводим заголовок
document.write("<h1>Это заголовок</h1>");
</script>
```

Многострочные комментарии начинаются с /* и заканчиваются */.

```
<script type="text/javascript">
/*
Код ниже выводит
один заголовок и параграф
*/
document.write("<h1>Это заголовок</h1>");
document.write("<p>Это параграф.</p>");
</script>
```

Операторы

- *арифметических действий,*
- *присваивания,*
- *инкрементные,*
- *декрементные.*

Оператор присваивания = используется, чтобы присваивать значения переменным JavaScript.

```
y=5;
    z=2;
    x=y+z;
// значение x после выполнения предложений выше
    будет равно 7.
```

Операторы

Пусть **x=10** и **y=5**, таблица объясняет операторы присваивания:

Оператор	Пример	Эквивалентная операция	Результат
=	x=y		x=5
+=	x+=y	x=x+y	x=15
-=	x-=y	x=x-y	x=5
*=	x*=y	x=x*y	x=50
/=	x/=y	x=x/y	x=2
%=	x%=y	x=x%y	x=0

Арифметические операторы используются для выполнения арифметических операций между переменными и/или значениями.

Пусть **y=5**,

Оператор	Объяснение	Пример	Результат	
+	Сложение	x=y+2	x=7	y=5
-	Вычитание	x=y-2	x=3	y=5
*	Умножение	x=y*2	x=10	y=5
/	Деление	x=y/2	x=2.5	y=5
%	Деление по модулю (остаток от деления)	x=y%2	x=1	y=5
++	Инкремент	x=++y	x=6	y=6
		x=y++	x=5	y=6
--	Декремент	x=--y	x=4	y=4
		x=y--	x=5	y=4

Операторы

Оператор **+** также может использоваться, чтобы соединять строковые переменные или текстовые значения друг с другом.

```
txt1="Какой сегодня";
txt2="прекрасный день";
txt3=txt1+txt2;
```

После выполнения

txt3 "Какой сегодняпрекрасный день".

Если вы складываете число и строку, результатом будет строка!

```
x=5+5;
document.write(x);
x="5"+"5";
document.write(x);
x=5+"5";
document.write(x);
x="5"+5;
document.write(x);
```

Операторы сравнения

используются в логических предложениях для определения равенства или неравенства между переменными или значениями.

Пусть **x=5**

Оператор	Описание	Пример
==	равно	x==8 это ложь x==5 это истина
===	точно равно (значение и тип совпадают)	x===5 это истина x==="5" это ложь
!=	не равно	x!=8 это истина
>	больше чем	x>8 это ложь
<	меньше чем	x<8 это истина
>=	больше или равно	x>=8 это ложь
<=	меньше или равно	x<=8 это истина

Как Это Можно Использовать

Операторы сравнения могут использоваться в условных предложениях, чтобы сравнивать значения и предпринимать определенные действия в зависимости от результата:

```
if (age<18) document.write("Слишком молодой");
```

Логические Операторы

используются для определения логических отношений между переменными или значениями.

Пусть **x=6 и y=3**,

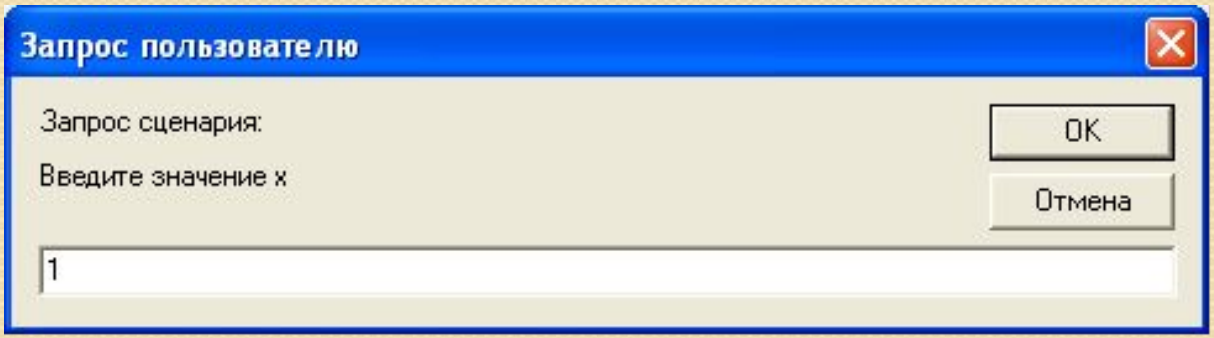
Оператор	Описание	Пример
&&	логическое И	(x < 10 && y > 1) это истина
||	логическое ИЛИ	(x==5 || y==5) это ложь
!	логическое НЕ	!(x==y) это истина

Задание

Вычислите значение выражения по формуле:

7*X + 10 / X – 8*X + 12.

В ходе работы использовать всплывающие окна «Alert» и «Prompt»



Условный Оператор

JavaScript также имеет условный оператор, который присваивает значение переменной на основе некоторого условия.

Синтаксис:

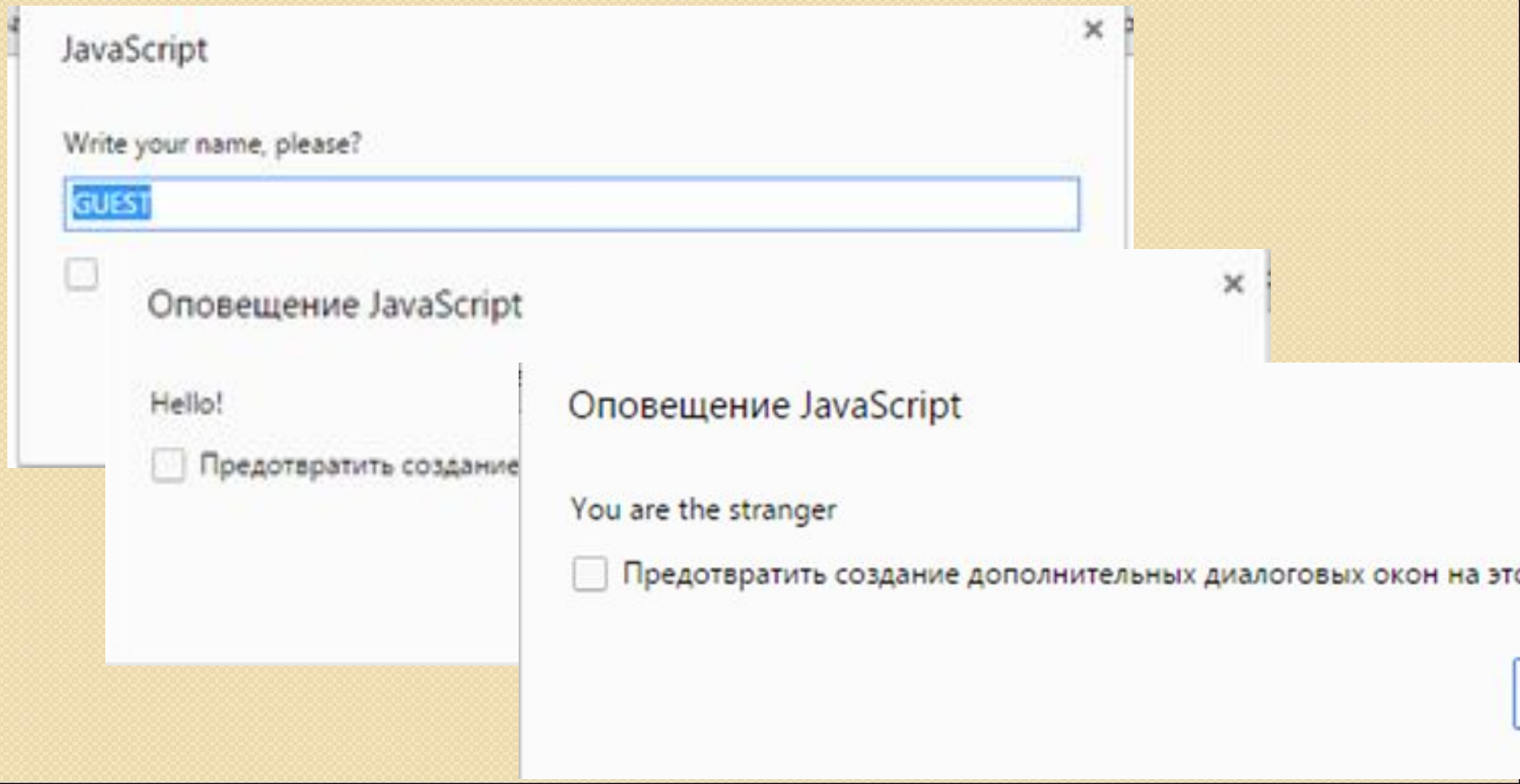
имя_переменной=(условие)?значение1:значение2

greeting=(visitor=="ВАСЯ")?"Привет Вася ":"Привет";

Если переменная **visitor** имеет значение "ВАСЯ", то переменной **greeting** будет присвоено значение "Привет Вася ", в противном случае ей будет присвоено значение "Привет".

Условный Оператор. Задание 1

Используя всплывающие окна «Alert» и «Prompt», проверить является ли пользователь - GUEST, если да - вывести сообщение «Hello!», иначе - сообщение «You are the stranger»



Условный Оператор. Задание 2

Используя всплывающие окна «Alert» и «Prompt», написать программу, которая по паролю определяет доступ к информации, если пароль – 111, то уровень доступа «А», если пароль – 222, то уровень доступа «В», иначе - - сообщение «You are the stranger»