GODOT. Подключаем JavaScript.

В этой статье покажу как подружить **Godot** со скриптами **Javascript**.

В рамках статьи я не буду концентрироваться на самом **javascript**, подразумевается, что если вы читаете эту статью то вы знакомы с джаваскриптом.

Первое что сделаем — добавим в проект **Label** и **Button**.
Соответственно будем нажимать на кнопку и либо отправлять какие то данные , либо получать.

И второе — подключим темплеит экспорта в HTML5.
(если у вас его нет — годо предложит вам его установить)

После этого у вас в верхнем меню добавится новая кнопочка.

Итак, всё готово для написания кода.

В годо для работы с **Javascript** кодом есть отдельный класс, он так и называется — **JavaScript**.

Как мы видим из описания, работает он только с HTML5 по понятным причинам. Ну и позволяет нам обрабатывать какие то скрипты или связываться со сторонними API (например ВК или Яндекс).

Подключаем новый скрипт. Подключаем сигнал с кнопки.
И для начала просто выведем какой то текст в Print.

Нажимаем на кнопку со значком **HTML5** затем **Run in Browser**, после чего у вас откроется ваш дефолтный браузер. Я использую Chrome, поэтому всё дальнеишее будет из него.

Если сейчас нажать на нашу кнопку, то наш принт выведет «dd» в консоль Chrome.
Открываем консоль.

Наблюдаем там наш текст.

Вроде пока всё работает. Теперь будем подключать джаваскрипт.
Для этого открываем экспортер, и находим окно **Head Include**.
Весь код из этого окна при экспорте добавляется в **html** фаил всего вашего проекта, и мы в последствии можем к нему обращаться.

Напишем простенький скрипт который будет содержать всего одну переменную.

Возвращаемся в скрипт годо. И вместо вывода в print пишем код для обработки уже непосредственно нашего джаваскрипт кода.

Для того чтобы получить весь код с html странички используется интерфеис **window**.
После чего мы просто берем значение нашей переменной **info** из джаваскрипта.

Запускаем, и видим что всё отработалось.

Мы так же можем выполнять функции в джаваскрипте из скрипта годо.
Для примера я вписал функцию которая возвращает нам значение переменной **some**.

И обращаюсь из годо уже непосредственно в эту функцию.

В итоге получаем результат.

В следующем примере рассмотрим такую штуку как колбеки.
Допустим у нас есть джаваскрипт, который срабатывает не моментально. И результат его работы нам надо передать в годо.
Для примера я написал скрипт который делает задержку в 2 секунды , и затем меняет нашу переменную test.

Для того чтобы принять результат этой функции, мы можем использовать метод create_callback().

Выносим колбек в переменную, после чего вызываем её в качестве аргумента при запуске функции из джаваскрипта.

Ну и собственно прописываем саму функцию которая будет вызвана в годо.

JS возвращает нам данные в виде массива. Поэтому тут берется нулевой индекс.

Теперь, запустив это всё в браузере можем увидеть результат работы этой всей связки.

В общем и целом это наверное всё.
Удачи !

GODOT. Про вызов функции в меню сигнала.

Обычно, когда мы подключаем сигнал в нашем скрипте появляется функция которая этот сигнал обрабатывает.
Для примера возьмем кнопку.

Подключим сигнал **pressed()** от кнопки в скрипт подвешенный на корневую ноду:

И получим в скрипте вот это :

Ну и далее мы можем чего то там понаписать внутри уже вот этой функции.

Но представьте, что у вас уже есть некая функция, которую вы хотите вызвать при нажатии на кнопку...

Для примера я накидал функцию **test**, и выводит она просто фразу через print:

В меню подключения сигнала просто вписываем её название :

Таким образом вызовется уже имеющаяся в нашем распоряжении функция.

Мы так же можем подключать сигналы вообще не имея подключенных скриптов. Но это касается только «родных» функций, ну и делать это нужно через Advanced меню.

Для примера сделаем так, чтобы кнопка удалялась по нажатию.

Заходим в меню подключения сигнала. Годо будет ругаться что нет скрипта.

Но это не беда.
Нажимаем Advanced, выделяем кнопку и пишем в строке метода queue_free

Ну и после нажимаем на Connect , и проверяем.
Кнопка будет удаляться при нажатии на неё.

Надеюсь принцип вы поняли. За сим всё, удачной разработки !