WLMill-Script

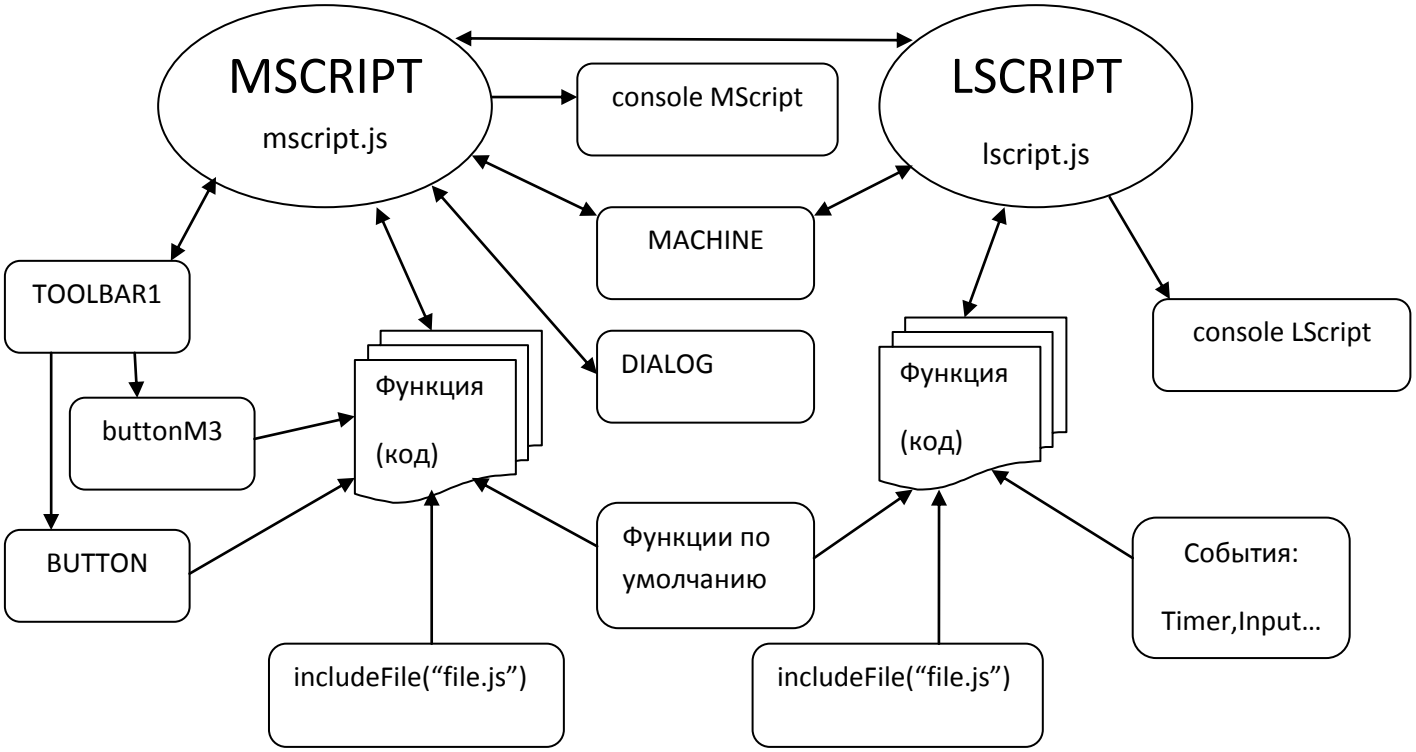
Содержание

1 Скрипты

В программе WLMill есть возможность выполнения функций-скриптов написанных на языке ECMAScript. Данный инструмент весьма полезен при работе с программой, с помощью функций-скриптов можно выполнять различные операции, которые небыли предусмотрены в стандартном функционале WLMill.

Язык ECMAScript достаточно прост, его описание можно найти в интернете, также существует большое количество видео с уроками по этому языку.

В WLMill есть два скрипта(машины выполняющие функции-скрипты).



У каждого скрипта есть очередь функций-скриптов которые будут выполняться последовательно.

Скрипты в WLMill:

- MScript – это скрипт который выполняет функции-скрипты в ходе работы WLMill. Например команды M3() M5() итд в G коде. Либо запуск функции-скрипта по нажатию на кнопку (userFunc1(),userFunc2()). Данный скрипт допускает задержки, остановки, ожидания в своих ыункциях.
- LScript – это фоновый скрипт который выполняет функции-скриптов из очереди в фоновом режиме. Данный скрипт нужен для организации быстрых обработок событий или вывода данных на выход. Скрипт LScript можно сравнить с обработчиком прерываний. Данный скрипт не допускает больших задержек.

По умолчанию базовые скрипты mscript.js и lscript.js. Должны находится в папке /wlmillconfig/script/ . Если их в ней нет, то будут созданы базовые файлы скриптов.

2 Функции и объекы скриптов.

В скриптах есть функции которые добавляются в очередь автоматически:

init()	Вызывается при инициализации.	MScript,LScript
M*()	Функция типа: M3, M4, M5 и тд. Вызов которых может происходить из G кода	MScript
ON()	Функция вызывается при нажатии на кнопку “On” в программе. *	MScript,LScript
OFF()	Функция вызывается при отжатие на кнопки “On” в программе.	MScript,LScript
PAUSE()	Вызывается после остановки выполнения программы.	MScript
CONTINUE()	Вызывается перед продолжением выполнения программы.	MScript
RESET()	Вызывается при сбросе – отмене всех текущих действий.	MScript
userFunc*()	Пользовательские скрипты для быстрого вызова. Пользователь может также установить свою иконку для кнопок быстрого вызова (устаревшие, работают)	MScript
changedInput(number,state)	Вызывается при изменении состояния входа number в состояние state.	LScript
changedOutput(number,state)	Вызывается при изменении состояния выхода number в состояние state.	LScript

*- Рекомендуется добавить в функцию ON(), скрипта MScript, вызов init(). – для обновления данных при включении станка

Также в скриптах доступны следующие элементы.

buttonM(3-9)	По нажатию происходит вызов функции M(3-9)()	MScript
buttonUserFunc(1-10)	По нажатию происходит вызов функции userFunc(1-10)()	MScript
MACHINE	Работа со станком	MScript,LScript
TIMER	Работа с таймерами	MScript,LScript
DIALOG	Вызов диалоговых окон	MScript
FILE	Работа с файлами	MScript,LScript
SCRIPT	Работа со скриптом.	MScript,LScript

TOOLBAR1(2)	Панели инструментов в WLMill.	MScript
MSCRIPT	Работа со скриптом из другого скрипта	LScript
LSCRIPT	Работа со скриптом из другого скрипта	MScript
GCODE	Доступ к текущим G кодам	MScript,LScript

2.1 Кнопки интерфейса button*(MScript)

Функция	Описание
bool isShow()	Возвращает 1 если кнопка видна
bool isEnabled()	Возвращает 1 если кнопка активна
bool isChecked()	Возвращает 1 если кнопка нажата (см setChekable)
void setShow(bool)	Установка видимости (отображения кнопки)
void setEnabled(bool)	Установка активности кнопки
void setChekable(bool)	Установка режима работы кнопки с фиксацией
void setChecked(bool)	Установка кнопки в положение нажато (см setChekable)
void setIcon(file)	Установка иконки кнопки. Вводится имя файла либо каталог относительно папки “wlmillconfig/icons” Для разделения каталогов используется такой слеш – “/”
void setIconFrom(file)	Установка иконки кнопки. Вводится имя файла полностью. (полный путь к файлу).
void setToolTip(txt)	Установка всплывающей подсказки
void setText(txt)	Установка текста на кнопку
void setShortcut(txt)	Установка клавиш быстрого вызова. Например "Ctrl+1"
void setScript(script)	Задаётся скрипт, который будет выполнятся при нажатии.
setScriptRelease(script)	При отжатие кнопки будет выполняться скрипт "script", если кнопка работает в режиме фиксации "setCheckable(1)"
int addButtonMenu(name,script)	Добавляется кнопка в сплывающее меню кнопки. Name – название кнопки. Script – скрипт выполняющийся по нажатию. Возвращает индекс текущей кнопки (0,1,2….)
void setIconSub(index,file)	Установка иконки на кнопку меню. Относительный путь от “wlmillconfig/icons”
void setIconFromSub(index,file)	Установка иконки на кнопку меню. Абсолютный путь.
void setToolTipSub(txt)	Установка всплывающей подсказки на кнопку меню.
void setTextSub(txt)	Установка текста на кнопку меню.
void clearMenu()	Удаление меню кнопки.
bool isVisible()	Возвращает 1 если кнопка видна (устарела 10.12.2021)
void setVisible(bool)	Установка видимости (отображения кнопки) (устарела 10.12.2021)

Пример установки функции скрипта на кнопку:

MYBUTTON.setScript(“M3()”) //по нажатию на кнопку запустится функция M3

MYBUTTON.setScript(“SCRIPT.console(‘press MYBUTTON’)”) //по нажатию на кнопку в консоль скрипта запишется ‘press MYBUTTON’.

2.2 MACHINE - основной элемент функционала (MScript,LScript)

Функция	Описание
bool getInput(index) bool getIn(index)	Возвращает текущее состояние входа
bool getOutput(index) bool getOut(index)	Возвращает текущее состояние выхода
setOutput(index,state) setOut(index,state)	Устанавливает выход в заданное состояние
setOutPulse(index,state,time)	Формирует импульс заданного состояния, и заданной продолжительности (в миллисекундах).
setOutTog(index)	Изменяет состояние выхода на противоположный
enableSOut(enable)	Активирует выход S
goDriveFind("*")	Запускает процедуру поиска положения двигателя по датчику
bool isDriveActiv("*")	Если двигатель активен (выполняет движение или выполняет автоматическую операцию) то возвращает true.
void setCurPositionSC("*", pos)	Устанавливает текущее значение заданной координаты в активной системе координат
bool runGCode(txt)	Выполняет G код
double getCurPositionSC("*")	Возвращает положение оси в текущей СК
double getCurPosition("*")	Возвращает положение оси в машинной СК G53
bool getInProbe()	Возвращает текущее состояние входа inProbe
bool isActiv()	Возвращает 1 если станок активен (есть задание на перемещение либо происходит перемещение)
void clearGProbe()	Очистка данных пробингов
void addGProbeXY(X,Y,Z,angle,distance, distanceA);	Добавление задания пробинга в плоскости XY. XYZ – координаты ожидания контакта angle – угол движения к контакту distance – расстояние до контакта distanceA – расстояние после контакта
void addGProbeZ(X, Y, Z, distance, distanceA);	Добавление задания пробинга в отрицательном направлении оси Z. XYZ – координаты ожидания контакта distance – расстояние до контакта distanceA – расстояние после контакта
void goGProbe()	Запуск процесса пробинга
double getGProbeSC(index,nameCoord)	Возвращает значение координаты в текущей СК. Для пробинга с индексом “index” и названием координаты “nameCoord”
double getGProbe(index,nameCoord)	Возвращает значение координаты в машинной СК. Для пробинга с индексом “index” и названием координаты “nameCoord”
bool isCompleteGProbe()	Возвращает 1 если все точки были найдены, и 0 если не все.
void setPause(set)	Поставить на паузу(1) или снять с паузы(set=0).
void stopMov()	Прекратить все движения.
bool loadGProgram(nameFile,show)	Загрузка программы nameFile. Произвести её отрисовку (show=1) или нет (show=0). Возвращает 1 в случае успеха и 0 если программа не была загружена.
bool runGProgram(ielement)	Запуск выполнения программы начиная с индекса

	ielement.
float getPercentSpeed()	Возвращает текущую корректировку скорости перемещения, в процентах
float getPercentSOut()	Возвращает текущую корректировку S - обороты шпинделя, в процентах
setPercentSpeed()	Устанавливает корректировку скорости, в процентах
setPercentSOut()	Устанавливает корректировку S - обороты шпинделя, в процентах

runMScript(txt)	Запуск M скрипта с учетом его запуска.
	Например, остановка шпинделя при паузе:
	PAUSE()
	{
	…
	MACHINE.runMScript("M5()");
	}
	Тогда после продолжения (отжатия кнопки «пауза») шпиндель запустится автоматически.

2.3 TIMER – таймеры (MScript)

Элемент таймер содержит в себе 32 независимых таймера. Которые могут использоваться при работе со скриптами.

Функция	Описание
start(index)	Запускает таймер
restart(index)	Перезапускает таймер
stop(index)	Останавливает таймер
long getCount(index)	Возвращает счётчик таймера (в мсек)

2.4 DIALOG - диалоги

Помимо всего прочего имеется интерфейс для общения с пользователем из скрипта. Работа с данным элементом строится по следующему принципу:

- создаётся окно (вызов соотв. диалога: enterNum(txt), enterString(txt) итд)
- обрабатывается данные из окна (при необходимости)

Функция	Описание
message(txt)	Вывод окна с сообщением.
question(txt)	Форма диалога с вопросом. Возвращает 1 в случае согласия или 0 в случае отмены.
num enterNum(txt,default=0)	Вывод окна с подсказкой для ввод числа. Txt – комментарий ввода Default – число при открытие окна(не обязательный параметр). Возвращает введенное число.
str enterString(txt,default="")	Вывод окна с подсказкой для ввод строки. Txt – комментарий ввода Default – строка при открытие окна (не обязательный параметр). Возвращает введенную строку.
str enterSaveFile(txt1,lastFile)	Вывод окна выбора сохранения файла. Возвращает путь к файлу.
bool isOk()	Была ли нажата кнопка подтверждения ввода
bool isShow()	Отображается ли какое либо окно (устарела 10.12.2021)
real getNum()	Возвращает введённое число (устарела 10.12.2021)
txt getString()	Возвращает введённую строку (устарела 10.12.2021)

void setString(txt)	Устанавливает строку в окно ввода строки (устарела 10.12.2021)
void setNum(numl)	Устанавливает число в окно ввода числа (устарела 10.12.2021).

Пример – ввода числа:

```
var num=DIALOG.enterNum(“введите число:”,0);
if(DIALOG.isOk())
 SCRIPT.console(“вы ввели:”+num);
else
 SCRIPT.console(“вы отменили ввод”);
```

2.5 FILE - работа с файлами (MScripr,LScript)

Элемент макросов FILE даёт возможность работы с файлами.

Функция	Описание
bool createFile(nameFile)	Запись пустого файла. (стирание файла)
bool write(nameFile,string)	Запись строки в файл. nameFile – имя файла string – строка
bool saveValue(nameFile,nameVal,value)	Сохранение переменной в файл типа “ini” nameFile – имя файла nameValue – имя переменной value – переменная (число или строка)
double loadValue(nameFile,nameVal,defValue)	Чтение переменной типа число из файла типа “ini” nameFile – имя файла nameValue – имя переменной defValue – значение по умолчанию
txt curPath()	Возвращает путь к исполняемому файлу (WLMill)

Пример чтения параметра из файла data.ini (расположен в “папка WLMill”/mydata/):

```
var value
value=FILE.loadValue(FILE.curPath()+”mydata/data.ini”,”myValue”,0) //
value=0 //если файла или переменной нет, будет присвоено значение по умолчанию
value=50 //если в файле “data.ini” записано:
// [GENERAL]
//myValue=50
```

Пример записи параметра в файл data.ini (расположен в “папка WLMill”/mydata/):

```
var speed=100

FILE.saveValue(FILE.curPath()+”mydata/data.ini”,”param/mySpeed”,speed)

//если в файле “data.ini” записано:

// [param]

//mySpeed=100
```

2.6 SCRIPT – работа со скриптом (MScripr,LScript)

Элемент макросов SCRIPT даёт возможность работы с скриптами

Функция	Описание
void console(txt)	Вывод текста в консоль скрипта.
void runFunction(func)	Добавление в очередь скрипта вызов функции-скрипта.
void runScript(script)	Добавление в очередь скрипта.

void includeFile(file)	Добавление файла в скрипт.
	File – файл в папке wlmillconfig/script/include
	Файлы должны иметь разрешение "js". Допустимо использование маски файлов для сокращения кода.
	
	Например:
	
	SCRIPT.includeFile("script.js") // загрузит файл "script.js" из папки "/wlmillconfig/script/"
	
	SCRIPT.includeFile("/include/script.js") // загрузит файл "script.js" из папки "/wlmillconfig/script/include"
	
	SCRIPT.includeFile("/include/*.js") // загрузит все файлы из папки "/wlmillconfig/script/include"
	
	SCRIPT.includeFile("/my*.js") // загрузит все файлы из папки "/wlmillconfig/script" по маске "*.js" (например myProbe.js, myFunc)

void includeFileFrom(file)	Добавление файла в скрипт.
	File – файл, полный путь
	Файлы должны иметь разрешение "js". Допустимо использование маски файлов для сокращения кода.
	
	Например:
	
	SCRIPT.includeFile("script.js") // загрузит файл "script.js" из папки "/wlmillconfig/script/"
	
	SCRIPT.includeFile("/include/script.js") // загрузит файл "script.js" из папки "/wlmillconfig/script/include"
	
	SCRIPT.includeFile("/include/*.js") // загрузит все файлы из папки "/wlmillconfig/script/include"
	
	SCRIPT.includeFile("/my*.js") // загрузит все файлы из папки "/wlmillconfig/script" по маске "*.js" (например myProbe.js, myFunc)

id setTimeout("script",time_ms)	По таймеру добавится в очередь скрипта функция-скрипт (script) через time_ms. Возвращает идентификатор таймера.
id setInterval("script",time_ms)	По таймеру будет периодически добавляться в очередь скрипта функция-скрипт (script) каждые time_ms. Возвращает идентификатор таймера.
void clearTimeout(id)	Удаляет задание таймера (setTimeout) по идентификатору. Если id не указано или 0. То будут удалены все таймеры данного типа.
void clearInterval(id)	Удаляет задание таймера (setInterval) по идентификатору. Если id не указано или 0. То будут удалены все таймеры данного типа.
bool isTimeout(id)	Проверка активен ли таймер по его id. 1-активен, 0- неактивен
bool isInterval(id)	Проверка активен ли таймер по его id. 1-активен, 0- неактивен
process()	Необходимо вызывать в долгих циклах ожидания.

После добавления файла с помощью includeFile (includeFileFrom). В скрипте также появляются глобальные переменные. Например

includeFile(“myscript.js”)

myscriptFileINI – полный путь до файла конфигурации скрипта, находится рядом с файлом myfile.js

myscriptPath - полный путь до папки файла скрипта

Пример добавления таймера с периодичьностью:

SCRIPT.setInterval(“MACHINE.setOutTog(4)”,1000) // каждую секунду будет переключение выхода 4

2.7 TOOLBAR1(2) - Панели инструментов в WLMill. (MScripr,LScript)

TOOLBAR1(2) – это две панели инструментов в WLMill. В которые можно добавлять новые кнопки.

Функция	Описание
void addButton(name,script)	Добавление новой кнопки с именем name(заглавными буквами). Имя name должно быть уникальным (не повторяться). После вызова появляется новый обект с таким же именем. После чего с ним можно работать как с элементом button.

Пример создание кнопки:

TOOLBAR1.addButton(“MYBUTTON”,”M5()”) // добавляется кнопка MYBUTTON в панель инструментов TOOLBAR1 и создаётся новый объект MYBUTTON

MYBUTTON.setIcon(“myicon.png”) // устанавливаем картинку на кнопку

Кнопки не удаляются!!! Возможно только при перезапуске WLMill;

2.8 GCODE – Доступ к текущим G кодам. (MScripr,LScript)

Функция	Описание
void push()	Сохранение текущего состояния G кода в буфер.
void pop()	Восстановление состояния G кода из буфера.

void setHTool(index,H)	Сохранение значения корректора для длины инструмента.
void setDTool(index,D)	Сохранение значения корректора для диаметра инструмента.
double getHTool(index)	Возвращает значение корректора длины инструмента
double getDTool(index)	Возвращает значение корректора диаметра инструмента
double getValue(name);	Возвращает текущее значение переменной G кода по имени.
double getGSC()	Возвращает номер текущей СК.
bool isGCode(number)	Возвращает 1 если текущий G код активен. 0 – неактивен
bool isMCode(number)	Возвращает 1 если текущий M код активен. 0 – неактивен

2.9 Примеры

2.9.1 Запуск G кода

```
MACHINE.runGCode("G0 X0 Y0")  //начинаем перемещение

MACHINE.runGCode("G1 X10 Y0")

MACHINE.runGCode("G1 X10 Y10")

MACHINE.runGCode("G1 X0 Y10")

while(WAIT(MACHINE.isActiv()));  //ожидаем завершение движений
```

2.9.2 Пробинг

Проббинг состоит из 3 основных этапов.

- Очистка прошлых данных
- Добавление задания пробинга
- Ожидание завершения и считывания полученных данных

Общий пример:

```
MACHINE.clearGProbe(); //очищаем данные прошлого пробинга

MACHINE.addGProbeXY(0,0,-2,0,10); //добавляем точку пробинга. Ожидаемая точка с координатами 0,0,-2 направление пробинга 0 градусов (вдоль оси X +) расстояние пробинга 10 . Индекс этой точки 0

MACHINE.goGProbe(); //запускаем проббинг

while(MACHINE.isActiv()); //ожидаем его завершения

var X = MACHINE.getGProbeSC(0,"X") //берём значении X

var Y = MACHINE.getGProbeSC(0,"Y") //берём значении Y
```