**Переменные**

Переменная – сигнал, именованная ячейка памяти, в которую можно записать или из которой можно прочитать значение. Переменные могут быть созданы при помощи ключевых слов **var**, **const**, **init**, **output**, **input** или автоматически, в соответствии с выражением, которое присваивается переменной, а также получены из Глобальных Параметров субмодели. Переменные могут иметь один из перечисленных ниже стандартных типов.

Идентификатор типа Описание типа данных

**integer** Целое значащее 32-битное число

**double** Действительное 64-битное число

**complex** Комплексное 128-битное число. Имеет формат вывода a±bi, где a – действительная часть, b – комплексная часть. Можно задать комплексное число также в виде (a,b)

**intarray** Массив (вектор-строка) целых чисел

**array** Массив (вектор-строка) действительных чисел

**carray** Массив (вектор-строка) комплексных чисел

**matrix** Матрица действительных чисел

**cmatrix** Матрица комплексных чисел

**boolean** Двоичное 1-битное значение, может иметь значение 0 или 1

**color** Цвет – эквивалентен типу **integer**

**point** Геометрическая точка – эквивалентна комплексному числу, имеет формат вывода (a,b), где a – абсцисса, b – ордината

**string** Строка символов

***Примечание.*** Далее под вектором понимается массив (вектор-строка) чисел, а вектор-столбец – это матрица, состоящая из нескольких строк и одного столбца.

**Пример.**

|  |  |
| --- | --- |
|  | **var**  aaa : **intarray** = [1,2,3], //массив целых чисел  bbb : **double**=555, //действительное 64 битное  qwe : **string**="qwe", //текстовое  ccc : **complex**=(1,2); //комплексное |

Кроме стандартных типов переменных могут быть переменные сложного типа – *записи*.

*Запись* декларируется при указании типа данных переменной в соответствии со следующим форматом:

<имя поля 1>{:<тип поля 1>}{=<начальное значение поля 1>}{…}

**Пример.**

|  |  |
| --- | --- |
|  | **var** newrec : (x = 0.0 , y = 0.0); |

***Примечание.*** Поля внутри декларации сложного типа определяются аналогично ключевому слову **var**.

Доступ к полю сложной переменной производится при помощи оператора « **->** » в соответствии со следующим форматом:

<имя записи> **->** <имя поля записи>

**Пример.**

|  |  |
| --- | --- |
|  | A = newrec**->**x; |

В блоке "Язык программирования" доступны следующие системные переменные:

|  |  |  |
| --- | --- | --- |
| Переменная | Тип данных | Назначение |
| **time** | Вещественное (double) | Модельное время |
| **stepsize** | Вещественное (double) | Шаг интегрирования |
| **goodstep** | Двоичное (Boolean) | Флаг «хорошего» шага |
| **getderi** | Двоичное (Boolean) | Флаг вычисления значений производных |
| **setstepflag** | Двоичное (Boolean) | Флаг принудительного присвоения шага |
| **newstepvalue** | Вещественное (double) | Принудительное значение шага при **setstepflag = 1** |

Переменная goodstep принимает значение 1, если вычисления производятся в конечной точке шага интегрирования, и при этом удовлетворяется условие заданной точности; во всех других случаях переменная goodstep принимает значение 0. Переменная getderi принимает значение 1 при вычислении матрицы Якоби посредством численного дифференцирования; во всех других случаях переменная goodstep принимает значение 0.

Специальные переменные доступные в графическом контейнере:

|  |  |
| --- | --- |
| Имя | Назначение и пример |
| **GROUPCOLOR** | Цвет фона группы |
| **GLOBALCOLOR** | Цвет фона главного окна |
| **PAINTSTEP** | Флаг, выставляемый при перерисовке изображения окна. |
| **CONTAINER\_NAME** | Имя графического контейнера, в котором написан скрипт. |
| **SCALEX** | К-т масштабирования контейнера по оси X |
| **SCALEY** | К-т масштабирования контейнера по оси Y |
| **KLINE** | К-т масштабирования линий внутри контейнера |
| **SNAPSTEP** | Шаг привязки объектов к сетке |
| **CHANGE** | Флаг принудительной перерисовки графического контейнера. |
| **PARENTOBJECT** | Ссылка на объект-владелец контейнера (для того чтобы изнутри контейнера осуществлять операции с объектом) |
| **REG\_MODAL\_FLAG** | Флаг, указывающий, что эту форму надо открывать модально в режиме управления по ссылке. Используется для создания панелей управления, работающих в модальном режиме – для этого в их секции инициализации надо установить REG\_MODAL\_FLAG = 1. |
| **old\_project\_name** | Имя предыдущего файла, загруженного в окно |
| **old\_link\_id** | Имя последнего идентификатора ссылки, загруженного в окно |
| **last\_active\_screen\_id** | screen\_id последнего активного окна |
| **screen\_id** | Идентификатор окна редактора – произвольная строка, присваиваемая в скрипте |
| **last\_active\_project\_id** | Ссылка окна последнего активного проекта |
| **SYSTEM\_DB\_ROOT** | Путь хранения различных файлов данных, установленный для графической оболочки. |
| **DYNAMIC\_PROPERTY\_ITEMS** | Переменная для указания через скрипт подсказки выбора (для скриптов внутри свойств блока) |
| **CENTER\_X** | Текущее смещение по оси Х видимого центра графического контейнера относительно стандартного центра |
| **CENTER\_Y** | Текущее смещение по оси Y видимого центра графического контейнера относительно стандартного центра |
| **last\_active\_hwnd** | Указатель на последнее активное окно (хэндл) |
| **restartstep** | Шаг сохранения рестартов |
| **time** | Модельное время |
| **submodel** | Указатель на блок-родитель субмодели |
| **prjinited** | Флаг, выставляемый в единицу после завершения инициализации проекта. Пока инициализация не завершена, флаг имеет значение ноль. |
| **prjinitializing** | Флаг, выставляемый в единицу в процессе инициализации модели на расчет. Когда инициализация завершена, флаг сбрасывается в ноль. |

Переменные могут задаваться при помощи ключевых слов **var**, **const, input, output, init.**