Динамические формы: Исчерпывающее руководство для новичков

Кирилл Кривошеев

9 марта 2019 г.

Оглавление

Глава 1

Введение

1.1 Запуск динамических команд

Одним из первых вызовов, который бросает динамический язык начинающему разработчику, является необходимость настройки рабочего окружения HP Service Manager для редактирования шаблонов динамических форм. Однако, благодаря мощным возможностям динамического языка, сделать первые шаги в разработке можно гораздо проще: достаточно открыть шаблон обращения «Исполнятор» в dev-среде портала «Лицо ДРУГа». Это простая динамическая форма, состящая всего из двух полей: «Команда» и «Результат».

1.2 Первая программа

По давней традиции, первая программа, которую начинающий разработчик пишет на новом языке, является «Hello, world!». На динамическом языке она выглядит так:

```
>>> setValue([Hello, world!],[result])
```

Для выполнения программы введите ее в поле «Команда» и переставьте курсор в поле «Результат». В итоге, в поле «Результат» вы увидите текст:

```
Hello, World!
```

Поздравляем! Вы сделали первый шаг на безумно интересном пути к освоению могучих возможностей создани интерактивных форм, который дает вам Динамический язык. Это пример команды **setValue**, которая устанавливает значение, указанное первым аргументом в квадратных скобках в поле, указанное во втором аргументе.

Здесь и далее, все команды, которые вводятся в поле «Команда» будут предваряться символами

```
>>>
```

За которыми следует результат выполнения в поле «Результат»

1.3 Чувствительность к регистру

Динамческий язык нечувствителен к регистру. Это означает, что программы:

```
>>> setvalue([Hello, world!],[result])
```

и

```
>>> setvalue([Hello, world!],[result])
```

и даже

```
>>> sEtVaLuE([Hello, world!],[result])
```

С точки зрения интерпретатора динамического языка являются индентичными и произведут одинаковый результат. И несмотря на то, что последний пример выглядит, безусловно, наиболее круто, в примерах к книге мы будем придерживаться так называемого CamelCase, так как он наиболее удобочитаем:

```
>>> setValue([Hello, world!],[result])
```

1.4 Арифметические функции

Перейдем от «Hello, world!» к арифметике. Как и любой, уважающий себя язык программирования, динамический язык предоставляет возможность работы с основными арифметическими операциями, а именно **plus**, **minus** и **mul** для сложения, вычитания и умножения, а также **div** для деления:

```
>>> setValue([plus([68],[1])],[result])
69
>>> setValue([minus([69],[1])],[result])
68
>>> setValue([mul([6],[7])],[result])
42
>>> setValue([div([84],[2])],[result])
42
```

На самом деле, указанные функции обладают гораздо большим набором возможностей, но для первого знакомства этого будет достаточно.

1.5 Значения и типы

1.6 Операторы ветвления

1.7 Отладка

1.8 Упражнения

Откройте шаблон «Исполнятор» и решите следующие задачи:

1. Сколько будет в градусах Цельсия температура в 69 Фаренгейт?
2. Сколько спринтов потребуется для вывода в промышленную эксплуатацию шаблона «Совместная поездка» если каждый спринт исправляется 6 багов, и обнаруживается 4 новых?