

И. В. Лошков,

Национальный исследовательский Московский государственный строительный университет,

Д. И. Лошков,

Университет штата Мэн, США

СТАНДАРТ ОПИСАНИЯ ПОЛЯ ВВОДА-ВЫВОДА В ПРОГРАММАХ ТЕСТИРОВАНИЯ

Аннотация

Широкое внедрение компьютерного тестирования требует стандартизации в этой области. В статье предлагается стандарт описания процесса вывода на экран монитора окон, посредством которых производится вывод вопросов и ввод ответов во время компьютерного тестирования. Стандарт использует формат передачи данных JSON.

Ключевые слова: компьютерное тестирование, компьютерная программа, стандарт, JSON, объект, свойства объекта, параметры элемента, форматирующая строка, поле ввода-вывода, управляющий элемент.

Введение

К настоящему времени разработано достаточно большое количество обучающих и тестирующих компьютерных программ. Для унификации и переносимости учебного материала, созданного разными коллективами, широкое распространение в таких программах получили два программных комплекса спецификаций и стандартов представления данных: SCORM [2] и IMS Common Cartridge [3]. Эти комплексы включают в себя ряд разработанных стандартов и реализующих эти стандарты программных модулей для представления различных компонент учебного материала. Однако оба указанных комплекса не включают в себя стандарты для представления сложных тестов, в то время как тесты являются важной частью учебного материала. Проблема стандартизации тестирования частично решена в комплексе IMS QTI (IMS Question & Test Interoperability) [4]. В этом комплексе реализованы многие способы введения ответов тестируемыми и оценки введенных ответов. Более подробно о проблеме стандартизации

при компьютерном обучении можно прочитать в статье И. А. Посова [1].

В процессе компьютерного тестирования на экране монитора выделяется область, на которой происходит взаимодействие человека и программы ЭВМ. Эту область будем называть *полем ввода-вывода*. Структура этого поля может быть достаточно разнообразной. В простейшем случае на этом поле размещаются окна, в которых предлагаются варианты ответов на тест, из которых необходимо выбрать правильные ответы. В более сложных тестах на этом поле могут располагаться окна ввода данных, в которые требуется вписывать найденные ответы на задачи теста. По мере усложнения тестовых задач и расширения сферы применения тестирования происходит усложнение взаимодействия тестируемого и программы ЭВМ. Так, в ходе ответа на заданный вопрос могут изменяться или появляться новые окна ввода или вывода, т. е. ответ на вопрос теста может иметь *интерактивный характер взаимодействия тестируемого с программой*. Соответственно, программирование поля сложной структуры становится важным элементом при написании тестовой задачи.

Контактная информация

Лошков Игорь Владимирович, канд. физ.-мат. наук, доцент Национального исследовательского Московского государственного строительного университета; *адрес:* 129337, г. Москва, Ярославское ш., д. 26; *телефон:* (499) 183-59-94; *e-mail:* loshkovigor@rambler.ru

Лошков Дмитрий Игоревич, аспирант Университета штата Мэн, США; *адрес:* University of Maine, Orono, Maine, USA, 04469; *e-mail:* dmitry.loshkov@maine.edu

I. V. Loshkov,

Moscow State University of Civil Engineering (National Research University),

D. I. Loshkov,

University of Maine, USA

STANDARD OF DESCRIBING IO FIELD IN TESTING PROGRAMS

Abstract

The widespread introduction of computer-based testing requires a standardization of the area. In the article a standardization of specification of output process of windows on computer display which is used to read questions and input answers during computer testing is proposed. JSON data format is used for standardization mentioned above.

Keywords: computer testing, computer program, standard, JSON, object, object properties, parameters of the element, formatting string, IO field, control element.

Для обеспечения переносимости содержания тестовой задачи можно использовать стандарты указанных выше программных комплексов [2, 3]. Обеспечить переносимость вариантов ответа на задачу в достаточно широком диапазоне способов ответа может программный комплекс [4]. Однако если потребуется создать сложную (разветвленную) структуру поля ввода-вывода, то это остается индивидуальной (не стандартизированной) задачей программирования. Причем перенос программного блока для создания полей из одной учебной программы в другую будет также требовать значительных усилий.

Значительное упрощение этих процедур для сложных (структурированных) полей ввода-вывода может достигаться за счет внедрения стандарта (формата) описания полей. В этом случае достаточно будет в каждой тестовой программе использовать модуль, который будет переводить записанную структуру поля на основании стандарта в тот вид, который совместим с программой тестирования.

Цель данной статьи — представить основы стандарта описания сложно структурированного поля ввода-вывода для программы компьютерного тестирования.

Предлагается следующее краткое обозначение стандарта:

- на русском языке — СОПВВ (стандарт описания поля ввода-вывода);
- на английском языке — SNIOF (Standard of the notation the input-output field).

Компоненты стандарта

Для описания размещения полей ввода-вывода предлагается использовать форматизирующую строку, которая представляет собой текстовую строку, составленную из элементов, разделяющих или группирующих символов, внедряемых текстов и комментариев.

Назначение элементов — описание объектов, создающих окна на поле ввода-вывода.

Окна могут быть видимые, невидимые, пустые, скрытые.

- *Видимые окна* имеют графическое представление на поле.
- *Невидимые окна* не имеют графического представления, но могут переводиться в состояние видимого окна, поэтому занимают определенное пространство на поле.
- *Пустые окна* не имеют никакого наполнения, но занимают определенное пространство на поле и предназначены для помещения в них в дальнейшем скрытых окон.
- *Скрытые окна* не имеют графического представления, не занимают места на поле и предназначены для размещения их в дальнейшем в пустые окна.

Укажем функциональное назначение окон на поле ввода-вывода. Это:

- вывод информации, необходимой для тестируемых;
- ввод информации самими тестируемыми;
- управление порядком вывода и размещения окон.

В соответствии с функциональным назначением окон, создаваемых на основании элементов, последние можно объединить в группы:

- группа элементов вывода;
- группа элементов ввода;
- группа элементов ввода-вывода;
- группа управляющих элементов.

Если элемент используется для создания окна вывода информации (вопроса тестируемому), которое одновременно используется для ввода информации (ответа тестируемым), мы его будем относить к группе ввода-вывода. Все элементы группы ввода-вывода обладают также функцией управления.

Все элементы имеют две части: название и список параметров. Название обрамляется в кавычки. Для названия могут использоваться следующие символы: латинские или кириллические буквы, цифры, тире, подчеркивание, тильда. Элементы, описывающие скрытые окна, должны дополнительно обрамляться слева и справа наклонной чертой /.

Перечислим достаточно очевидные назначения окон.

Элементы вывода могут обеспечивать создание окон для вывода:

- текста в соответствии со стандартом используемого языка программирования;
- текста «как есть» — без изменения программой;
- формулы в научном виде;
- графического файла.

Элементы ввода могут обеспечивать создание окон для ввода:

- символов с клавиатуры, составляющих содержание ответа;
- формулы в указанном стандарте;
- исполняемого кода в указанном стандарте.

Элементы ввода-вывода могут обеспечивать создание окон:

- для выбора одного или нескольких ответов из выпадающего списка с вариантами ответов или инициирования новых элементов на основании выбранных вариантов ответа;
- для выбора одного или нескольких ответов из нескольких заданных вариантов ответа или инициирования новых элементов на основании одного или нескольких выбранных вариантов ответа;
- для выбора ответа из галереи графических файлов или инициирования новых элементов на основании выбранных вариантов ответа.

Управляющие элементы могут создавать окна для:

- размещения кнопки, управляющей размещением окон;
- резервирования места для вставки окна.

Разделяющими символами форматизирующей строки являются запятые и двоеточия. Запятые используются для отделения друг от друга элементов или блоков элементов, двоеточия — для отделения в параметрах элемента наименования и значения.

Группирующими символами форматизирующей строки являются: пара круглых скобок — (), пара прямых скобок — [], пара фигурных скобок — {}.

Внедряемый текст создается с помощью следующей структуры:

/# <переменная> #/,

где <переменная> — набор следующих символов без пробелов: латинских букв, цифр, подчеркивания, доллара. Модуль тестовой программы до начала разбора форматированной строки на основании содержимого <переменная> должен сформировать текст и подставить его вместо указанной выше структуры.

Комментарий может встречаться внутри форматированного текста в любом месте. Он обрамляется в начале прямой наклонной чертой и звездочкой, а в конце звездочкой и наклонной чертой. Пример: /*Это комментарий.*/

Запись элементов в форматированной строке

Все элементы форматированной строки записываются в составе блоков с помощью группирующих символов — пары круглых или прямых скобок. Сгруппированные элементы или блоки, размещенные внутри скобок, отделяются друг от друга запятыми. Блок может состоять и из одного элемента.

Пример записи блока-строки, использующего круглые скобки: $(A_1, A_2, A_3, \dots, A_n)$. Пример записи блока-столбца, использующего прямые скобки: $[A_1, A_2, A_3, \dots, A_n]$. Здесь в примерах $A_1, A_2, A_3, \dots, A_n$ — элементы.

При использовании круглых скобок окна соответствующих элементов на поле ввода-вывода будут располагаться в строку. При использовании прямых скобок — столбцом.

Вместо некоторых элементов в записи блока может находиться блок, который становится внутренним блоком. Если в записи форматированной строки встречаются блоки, создаваемые как с помощью круглых, так и с помощью прямых скобок, то соответствующие элементам окна будут располагаться на поле ввода-вывода в виде двухмерной структуры.

В такой двухмерной структуре могут быть следующие способы размещения окон: вложенный, построчный, постолбцовый.

- *Вложенное размещение.* Блок элементов и элементы создают в поле ввода-вывода каждый свое окно, одинаково равноценное в соответствии с записью того блока, к которому они принадлежат.
- *Построчное размещение.* Все поле разбивается на строки, и окно каждого элемента блока-столбца размещается в новой строке поля.
- *Постолбцовое размещение.* Все поле разбивается на столбцы, и окно каждого элемента блока-строки размещается в новом столбце.

Пример записи форматированной строки, содержащей объединение элементов с помощью разных скобок:

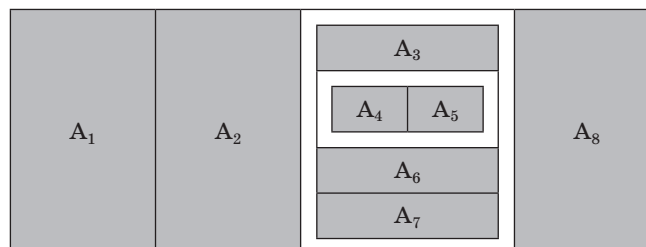
$(A_1, A_2, [A_3, (A_4, A_5), A_6, A_7], A_8)$,

где $A_1, A_2, A_3, \dots, A_n$ — только элементы.

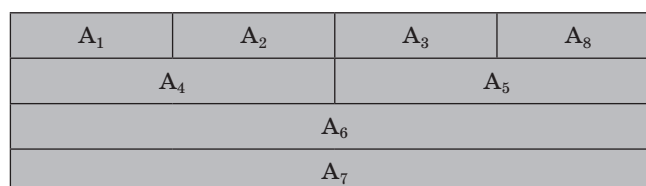
В результате применения написанной в этом примере форматированной строки программа должна сформировать двухмерную структуру размещения окон на поле ввода-вывода.

Все три варианта размещения окон изображены графически:

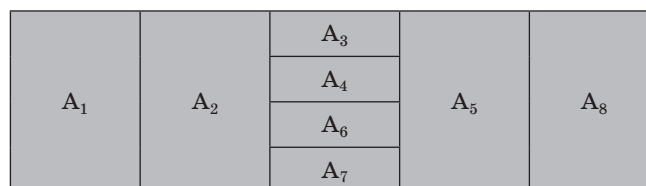
- для вложенного размещения:



- для построчного размещения:



- для постолбцового размещения:



Здесь A_1 — A_8 — окна, сформированные с помощью одноименных элементов.

Способ размещения окон на поле ввода-вывода определяется параметром блока (см. использование параметров в следующем разделе).

Элемент ввода (вывода) создает внутри ячейки окно ввода (вывода). Управляющий элемент создает внутри ячейки окно управления. Щелчок мыши внутри окна управления вызывает определенные действия.

Примечание. Элементы, соответствующие скрытым окнам, могут размещаться в любом блоке элементов и никак не влияют в момент создания поля ввода-вывода на его вид.

Запись параметров элементов или блоков в форматированной строке

Любой элемент описывает объект с набором свойств. Свойства объекта задаются с помощью параметров элемента. Кроме этого параметры элементов должны задавать размещение окон. Каждый параметр имеет название и значение. Параметры могут быть обязательные или факультативные (необязательные). Блоки элементов могут иметь параметры, которые относятся ко всем элементам этого блока. В частности, один из таких параметров может задавать один из способов размещения окон (при отсутствии такого параметра должен использоваться вложенный способ размещения).

Параметры элемента или блока задаются с помощью текста следующей структуры:

$$A_n : \{B_1 : C_1, B_2 : C_2, B_3 : C_3, \dots, B_k : C_k\}$$

Здесь:

A_n — название элемента или блока элементов;

$B_1, B_2, B_3, \dots, B_k$ — название параметра, обрамленное в кавычки;

$C_1, C_2, C_3, \dots, C_k$ — значение параметра.

Для элемента можно использовать конструкцию $A_n : \{$ без заполнения пространства между фигурными скобками, т. е. без задания параметров.

Для названия или значения параметра могут использоваться любые символы, допустимые в формате обмена данными JSON (JavaScript Object Notation) [5].

Значениями параметров могут быть простые величины (текст, число, дата, логическое значение, null и др.), массивы или объекты. Простые величины обрамляются в кавычки (цифры и логические значения могут не обрамляться кавычками). Массивы или объекты должны записываться в формате (стандарте) JSON. С помощью этого формата массив описывается следующей текстовой строкой:

["Значение 1", "Значение 2", ..., "Значение n"].

В результате применения этой строки создается одномерный массив с количеством элементов n . Элементы этого массива последовательно принимают значения: "Значение 1", "Значение 2", ..., "Значение n", причем первый элемент массива имеет индекс 0.

Согласно формату JSON объекты описываются следующей текстовой строкой:

{"Key 1" : "Значение 1", "Key 2" : "Значение 2", ..., "Key n" : "Значение n"}

В результате применения формата JSON создается объект со свойствами Key 1, Key 2, ..., Key n. Значения этих свойств будут соответственно: "Значение 1", "Значение 2", ..., "Значение n".

Все параметры можно разделить на две группы:

- не определяющие размещение окон;
- определяющие размещение окон.

Параметры, определяющие размещение окон, должны обеспечивать:

- задание вида размещения окон;
- задание места размещения (выше, ниже, правее, левее) новых окон относительно существующих;
- задание количества повторений однотипных окон;
- задание ориентации (вертикальной, горизонтальной) размещения повторяющихся однотипных окон;
- скрытие или невидимость окон и их проявление с помощью управляющих окон.

Заключение

В данной работе предложены основы стандарта описания поля ввода-вывода. Однако мы не коснулись в ней стандартизации названий и параметров элементов строки форматирования. Это составляет предмет дальнейшей стандартизации. Авторы составили предварительную концепцию развития этого стандарта и на ее основе создали программу, которая переводит форматизирующую строку в программный код, создающий поле ввода-вывода на интернет-странице.

Список использованных источников

1. Посов И. А. Стандарты представления учебных заданий в системах дистанционного обучения // Компьютерные инструменты в образовании. 2013. № 6. <http://ipo.spb.ru/journal/index.php?article/1611/>
2. Advanced Distributed Learning (ADL). Sharable Content Object Reference Model (SCORM) 2004 / пер. с англ. Е. В. Кузьминой. М.: ФГУ ГНИИ ИТТ «Информатика», 2005. <http://www.edu.ru/db/portal/e-library/00000053/SCORM-2004.pdf>
3. IMS Common Cartridge Specification // IMS Global Learning Consortium. <https://www.imsglobal.org/cc/index.html>
4. IMS Question & Test Interoperability Specification // IMS Global Learning Consortium. <https://www.imsglobal.org/question/index.html>
5. The JSON Data Interchange Standard. 1st Edition // Standard ECMA-404. October 2013. (ECMA International). <http://www.ecma-international.org/publications/files/ECMA-ST/ECMA-404.pdf>

НОВОСТИ

Квалифицированный труд станет общедоступной услугой

Многие работы, выполняемые сегодня квалифицированными специалистами, благодаря распространению технологий искусственного интеллекта и машинного обучения превратятся в общедоступные услуги, полагают аналитики Gartner. Влияние технологий искусственного интеллекта будет проявляться по-разному в зависимости от особенностей конкретных отраслей, организации работы и клиентуры. В некоторых отраслях искусственный интеллект будет

вытеснять людей, но в других автоматизация регулярных и монотонных задач позволит освободить время сотрудников для повышения уровня обслуживания, решения более сложных задач и даже поможет им снизить уровень стресса. Однако слишком высокий уровень автоматизации может лишить компанию гибкости, и поэтому следует задуматься о том, как сбалансировать применение технологий искусственного интеллекта и умений сотрудников.

(По материалам международного компьютерного еженедельника «Computerworld Россия»)