

Каменева Анастасия Евгеньевна

магистрант

Горбунова Алина Викторовна

студентка

Каменева Галина Анатольевна

канд. пед. наук, доцент

ФГБОУ ВО «Магнитогорский государственный

технический университет им. Г.И. Носова»

г. Магнитогорск, Челябинская область

КАСТОМИЗАЦИЯ МЕХАНИЗМА ВВОДА ТЕКСТА ДЛЯ WINDOWS PHONE 8.1

***Аннотация:** в статье рассматривается возможность кастомизации механизма ввода текста для платформы Windows Phone 8.1, а именно, создание альтернативы системной клавиатуре. Приведен обзор стандартного поля текстового ввода TextBox, некоторых особенностей и возможностей этого элемента. Обозначено также понятие «фокус ввода», названы элементы управления, которые обладают возможностью принимать на себя фокус, описаны условия, выполнение которых необходимо для того, чтобы фокусирование на этих элементах было доступно. Сформулированы рекомендации по созданию настраиваемой, локальной для приложения клавиатуры, поведение и функции которой максимально приближены к общепринятым.*

***Ключевые слова:** windows phone 8, textbox, клавиатура, поле ввода, фокус ввода.*

В среде мобильной разработки.NET для Windows Phone 8.1 (далее WP8.1) может возникнуть вопрос о кастомизации полей ввода и клавиатур. Чаще всего эта проблема возникает при разработке финансовых приложений для повышения безопасности использования.

Далее рассмотрим стандартный элемент управления `TextBox` и некоторые его особенности, позволяющие кастомизировать механизм ввода под определенные нужды, в частности, разберем понятие фокус на элементе.

`TextBox` – элемент управления, который используется для отображения текста, состоящего из нескольких строк в одном формате [1, с. 64].

Этот элемент обладает рядом довольно очевидных методов, событий и свойств, позволяющих изменить его внешний вид (цвет и размер шрифта и фона, цвет и толщина рамки и т. д.) и настроить стандартное взаимодействие с системой (события `Tapped`, `TextChanged`, `Paste` и т. д.), на которых не будем останавливаться подробно.

`InputPane` – это элемент пользовательского интерфейса, который отображается, когда пользователь выполняет действие, требующее ввода информации, например касается поля поиска или поля ввода в форме.

По сути, это контейнер для нативной клавиатуры. Мы можем узнать его расположение на экране, скрыть его или показать, подписаться на события `Showing` (возникает, когда панель появляется) и `Hiding` (когда панель скрывается). Прямой доступ к системной клавиатуре получить невозможно.

В рамках системы WP8.1 нельзя создать собственную клавиатуру, которая будет полностью заменять системную. Однако, для узкого ряда задач можно внутри конкретного приложения создать собственную клавиатуру и настроить `TextBox` таким образом, чтобы взаимодействие с такой клавиатурой было максимально приближено к «естественному», системному. Для этого необходимо рассмотреть такой аспект системы, как фокус ввода.

Фокус ввода – концептуальное понятие в построении графического пользовательского интерфейса, означающее наличие у определённого элемента исключительного права принимать клавиатурный ввод.

В WP8.1 иметь фокус могут следующие элементы: `TextBlock`, `Button`, все поля ввода (`TextBox`, `PasswordBox` и т. д.). Кроме того, чтобы эти элементы могли приобретать фокус, должны выполняться следующие условия:

- элемент должен быть активным, т.е. значение свойства `IsEnabled` должно быть `true`;
- элемент должен быть видимым, т.е. значение свойства `Visibility` должно быть `Visible`;
- элемент должен быть включен в `tab`-навигацию, т.е. значение свойства `IsTabStop` должно быть `true`.

Отсюда следует следующее: для того, чтобы отключить фокусирование элемента, но при этом оставить его активным, достаточно установить свойство `IsTabStop` равным `false`.

Рассмотрим некоторые события методы и свойства, связанные с фокусом `TextBox` (табл. 1).

Таблица 1

События, методы и свойства элемента `TextBox`

		Описание
События	<i>GotFocus</i>	Возникает, когда <code>TextBox</code> получает фокус.
	<i>LostFocus</i>	Возникает, когда <code>TextBox</code> теряет фокус.
Методы	<i>Focus</i>	Пытается перевести фокус на элемент.
Свойства	<i>FocusState</i>	Получает значение, указывающее, имеет ли данный элемент управления фокус и режим, в котором был получен фокус.
	<i>PreventKeyboardDisplayOnProgrammaticFocus</i>	Возвращает или задает значение, указывающее экранной клавиатуре, отображается ли она, когда элемент управления получает фокус программными средствами.

Рассмотрим подробнее возможные значения свойства `FocusState`. Перечисление `FocusState` имеет следующие значения, представленные в таблице 2.

Таблица 2

Значения перечисления `FocusState`

Член	Значение	Описание
<i>Unfocused</i>	0	Элемент в данный момент не имеет фокуса.
<i>Pointer</i>	1	Элемент получил фокус в результате действия указателя.
<i>Keyboard</i>	2	Элемент получил фокус с клавиатуры.
<i>Programmatic</i>	3	Элемент получил фокус с помощью преднамеренного вызова метода <i>Focus</i> .

Focus принимает значение типа `FocusState` и возвращает значение `true`, если фокус был установлен на элемент управления или уже находился на нем; значение `false`, если элемент не может получить фокус.

Нужно отметить, что невозможно убрать фокус с элемента управления путем вызова метода `Focus` со значением `FocusState.Unfocused` в качестве параметра – это вызовет исключение.

Исходя из всего вышеперечисленного можно сформулировать основные рекомендации, которых стоит придерживаться при проектировании и разработке собственного метода ввода. Создавая кастомную клавиатуру, разумно будет устроить ее как панель с набором кнопок (`Button`), которые не могут забирать на себя фокус. Тогда по нажатию на кнопку клавиатуры можно обновлять свойство `TextBox.Text` – добавлять туда необходимый символ, при этом фокус будет оставаться у элемента `TextBox`, что максимально похоже на системное поведение. Тогда можно настраивать события `GotFocus` и `LostFocus` желаемым образом, т.к. они будут возникать тогда же, когда бы они возникали при использовании системной клавиатуры. Кроме того, для использования кастомной клавиатуры необходимо скрывать системную. Один из вариантов решения этой проблемы – вызывать `InputPane.GetForCurrentView().TryHide()` в событии `GotFocus` – тогда при получении фокуса полем ввода системная клавиатура сразу же будет скрываться. Так же, если необходимо отдать фокус `TextBox` программным образом, логично будет установить свойство `TextBox.PreventKeyboardDisplayOnProgrammaticFocus` в значение `true` и вызывать функцию `TextBox.Focus` с параметром `FocusState.Programmatic` – тогда системная клавиатура не будет вызываться.

В зависимости от конкретных нужд при создании собственной клавиатуры можно комбинировать использование представленных выше методов и свойств.

Список литературы

1. Культин Н.Б. Основы программирования в Microsoft Visual C# 2010. – СПб.: БХВ-Петербург, 2011. – 368 с.