**UITextInputTraits**

**UITextContentType** позволяет разработчикам описывать ожидаемую «поддержку», предоставляемую системой, например:

* Автоматическое использование заглавных букв, проверка орфографии, умные кавычки, умные тире и умные пробелы (при вставке текста)
* Какая программная клавиатура и клавиша возврата должны отображаться, в том числе светлая или темная клавиатура
* Если клавиша «возврат» должна автоматически включаться / отключаться в зависимости от поля, имеющего содержимое
* Если поле представляет собой «защищенный» текстовый ввод (где содержимое заменяется на - s)
* Правила ввода пароля для поля ввода пароля, если запрашивается автозаполнение

**Autocapitalization Type (UITextAutocapitalizationType)**

**Должно ли это текстовое поле пытаться автоматически вводить слова с заглавной буквы, и если да, то как? Обратите внимание , что некоторые клавиатуры не поддерживают эту функцию, а именно: numberPad, phonePadиnamePhonePad.**

Существует 4 типа автоматического использования заглавных букв:

1. **Нет** : не пытайтесь извлечь выгоду для пользователя. Это полезно при запросе элементов, которые обычно не пишутся с заглавной буквы, таких как адреса электронной почты или любое место, где текст может быть «стилизован» пользователем.
2. **Слова** : старайтесь писать каждое новое слово с большой буквы. Это полезно при запросе таких вещей, как имя / фамилия или полное имя, адрес или титул.
3. **Предложения** : попытка использовать новые предложения с заглавной буквы, что является обычным случаем (на английском языке) для написания длинного текста.
4. **Все** : КРИЧАНИЕ ПОЛНОСТЬЮ ЗАГЛАВНЫМ ВВОДОМ! Полезно для работы с мэйнфреймами, штатов США и других сокращений, начинающихся с заглавных букв.

**Autocorrection Type (UITextAutocorrectionType)**

Если это текстовое поле пытается исправить ошибки от имени пользователя. Возможно, вы захотите внести исправления в «Биографию» пользователя, но это может раздражать пользователя, если попытается автоматически исправить URL-адрес, имя или адрес электронной почты их веб-сайта.

Значение по умолчанию - **Default** , что означает, что они включены, если они поддерживают текущую систему сценариев.

Есть 3 значения:

1. **По умолчанию** : Обычно это то же самое, что и включено, однако некоторые системы сценариев не поддерживают автокоррекцию.
2. **Вкл .** : включить автокоррекцию, если она доступна.
3. **Off** : отключить автокоррекцию.

### Smart Quotes, Smart Dashes and Smart Insert/Delete

Эти параметры касаются замены персонажей. Перечислений для **UITextSmartQuotesType**, **UITextSmartDashesType** и **UITextSmartInsertDelete** все по существу то же самое: **default, no и yes**.

1. **Умные цитаты** будут (по существу) принимают «прямые» кавычки **“** и **‘** и превратить их в охотницы, изогнутыми компаньонов: **“quote”** и **‘quote’** (Это чувствительно к региону , что текущий пользователь установил в их среде, и могут вести себя по- разному в зависимости от того, что их регион есть.)
2. **Умные тире** превратят несколько тире в более длинные односимвольные тире. Два тире превращаются в длинное тире, а три тире - на длинное тире. (есть много тире ... **en** и **em** тире, а также **bar**, если это вас интересует, проверьте страницу википедии: [Dash - Wikipedia](https://en.wikipedia.org/wiki/Dash" \t "_blank) ). Эта черта *является* влияние типа клавиатуры , когда установлен **default**.
3. **Интеллектуальная вставка / удаление** Во время событий вырезания, копирования и вставки система может добавлять или удалять пробелы. Эта черта *является* влияние типа клавиатуры , когда установлен **default**.

### Spell Checking Type (UITextSpellCheckingType)

Должно ли это текстовое поле пытаться исправить орфографию для пользователя? Похоже на автокоррекцию, но не совсем то же самое. Проверка орфографии покажет красные линии под словами с ошибками и позволит пользователю нажать на слово, чтобы исправить его.

Значение по умолчанию - **По умолчанию,** что включает проверку орфографии, если включена автокоррекция.

1. **По умолчанию** : Включено, если также включена автокоррекция.
2. **Вкл .** : включить проверку орфографии, если она доступна.
3. **Выкл .** : отключить проверку орфографии.

**UIKeyboardType**

Перечисление является очень простым интерфейсом для очень сложного понятия - какого рода программного обеспечения клавиатура лучших поддерживает ввод текста , что это поле ожидает. Здесь есть несколько важных предостережений, на которые стоит обратить внимание:

* **Установка этого значения в текстовом поле на самом деле не ограничивает ввод, который пользователь может попытаться ввести**, и должна рассматриваться только как вспомогательная функция. Например, поле «сумма в долларах» может использовать десятичный ввод, но это не означает, что аппаратная клавиатура по-прежнему не может вводить текст. Проверка окончательного ввода по-прежнему важна!
* В доступных типах клавиатуры есть «дыры», и было бы разумно рассмотреть возможность тестирования всех поддерживаемых языков и устройств для данного приложения, чтобы убедиться, что происходят разумные вещи. Видный, и часто расстраивает пример: IPad не поддерживает любые цифры только клавиатуры, а вместо этого использует «цифры и знаки препинания» клавиатуры для номера, номера телефона и десятичного прокладочных материалов. Другой, более интересный пример: ввод с цифровой клавиатуры может позволить пользователю вводить символы, отличные от 0-9, если в их текущем языке устройства используется другая система счисления. **asciiCapableNumberPad**. Однако, не ограничивает ввод только цифр ASCII (0-9).
* Использование **namePhonePad** и **phonePad** входы заставят пользователь использовать встроенные клавиатуры, и отключить любые пользовательские клавиатуры они установили для этой области. Вы также можете отключить / все / настраиваемые клавиатуры (если действительно необходимо) с помощью метода делегата приложения:

application(\_:shouldAllowExtensionPointIdentifier:)

Итак, когда и как следует использовать каждый тип клавиатуры? Трудно сказать с абсолютной уверенностью, потому что каждое приложение индивидуально, но мы можем, по крайней мере, посмотреть на поведение каждой клавиатуры и предложить несколько общих предложений:

1. **По умолчанию** - это «По умолчанию для текущего языка», а не какая-то конкретная, уникальная клавиатура по умолчанию, поэтому не делайте предположений по этому поводу. Обычно это будет похоже на то, как может выглядеть аппаратная клавиатура для данного языка и локали.
2. **Возможность ASCII** - максимально приближенная к «стандартной клавиатуре по умолчанию», насколько вы можете получить - это будет клавиатура, которая позволяет использовать только символы ASCII, без диакритических знаков, без цифр, кроме 0-9, и так далее. Запустите man asciiна своем локальном компьютере, чтобы увидеть таблицу кодов ASCII. Это полезно, если вы работаете с некоторым конкретным значением, которое должно быть только ASCII (например, старые серверные ВМ не поддерживают UTF-8, или, возможно, серийный номер, который может состоять только из ASCII AZ и 0-9). Не используйте это для таких вещей, как имена, если в этом нет крайней необходимости - существует большой мир, и пользователи хотят указать свое настоящее имя, а не какое-то близкое приближение.
3. **Цифры и знаки препинания**. Эта клавиатура именно то, что написано: клавиатура, на которой заметно выделяются цифры и знаки препинания. По сути, происходит Defaultпрямое переключение на страницу с пунктуацией клавиатуры.
4. **URL** - Ориентирована входящих URL - адресов, эта клавиатура поверхности **.**, **/** и **.com** где пробел будет (так как вы не должны действительно входить не-закодировать пробелы в URL в любом случае). На странице символов / знаков препинания также есть более ограниченный набор ключей, ориентированных на то, что вы обычно найдете в действующем URL-адресе. Используйте это, если вы ожидаете, что пользователь введет истинный, правильно отформатированный URL-адрес (например, ссылку на веб-сайт вашего пользователя).
5. **Веб-поиск** - версия URL-клавиатуры с меньшими возможностями, цель здесь - помочь с вводом URL-адреса или поискового запроса (как вы видите в Safari по умолчанию). Удерживая нажатой клавишу с точкой, вы получите автозаполнение для доменов верхнего уровня (.com, .net и других). Используйте эту клавиатуру, как это делает Safari, для мест, где пользователь может ввести поисковый запрос или URL-адрес.
6. **Адрес электронной почты** - также делает то, что написано - упрощает ввод адреса электронной почты с помощью символов @ и. клавиши, а также предоставление опций TLD при длительном нажатии на период. Также предоставляет меньший набор специальных символов, аналогично клавиатуре URL.
7. **Twitter** - Twitter сделал хэштег настолько популярным, что получил особый стиль клавиатуры, который выводит на передний план символ фунта / хеша / octothorp, а также знак @. Полезно для контекстов социальных сетей, где часто используются @ и #.

**Keyboard Look**

Это «светлая» или «темная» клавиатура, если таковая имеется. Apple предлагает использовать .dark для ввода уровня предупреждений, но не особенно четко или сильно. По умолчанию всегда light.

**Return Key Type**

Как должна выглядеть и говорить клавиша "возврат". Есть целый ряд из них, но многие из них были «упростили» далеко (например, yahooи google варианты будут показывать «Поиск»)

Мы уже смотрел на **UITextContentType** поле, так что конечный пункт рыть , чтобы также наиболее сложные: **UIKeyboardType**.

**Enables Return Key Automatically**

Делает то, что говорит, и (если установлено значение true) отключает клавишу возврата, пока пользователь не введет какие-либо данные. Однако это не мешает пользователю использовать клавишу возврата на аппаратной клавиатуре и является лишь улучшением пользовательского интерфейса для программной клавиатуры.

**Secure Text Entry**

Стоит ли скрывать данные ввода текста от других людей, которые также могут видеть экран? Чаще всего используется поле пароля, но и другие конфиденциальные данные также стоит скрыть по умолчанию.

**UITextFieldDelegate**

Набор дополнительных методов для управления редактированием и проверкой текста в объекте текстового поля.

**Декларация**

**@MainActor protocol UITextFieldDelegate**

**Обзор**

Текстовое поле вызывает методы своего делегата в ответ на важные изменения. Эти методы используются для проверки текста, который был введен пользователем, для реагирования на определенные взаимодействия с клавиатурой и для управления общим процессом редактирования. Редактирование начинается незадолго до того, как текстовое поле становится первым респондентом и отображает клавиатуру (или назначенное ему представление ввода). Процесс редактирования выглядит следующим образом:

1. Прежде чем стать первым респондентом, текстовое поле вызывает метод своего делегата . Используйте этот метод, чтобы разрешить или запретить редактирование содержимого текстового поля **.textFieldShouldBeginEditing(\_:)**
2. Текстовое поле становится первым респондентом.

В ответ система выводит на экран клавиатуру (или вид ввода текстового поля) и сообщения о и уведомления в случае необходимости. Если клавиатура или другое окно ввода уже были видны, система вместо этого отправляет уведомления и **.keyboardWillShowNotification**

**.keyboardDidShowNotification**

**.keyboardWillChangeFrameNotification**

**.keyboardDidChangeFrameNotification**

1. Текстовое поле вызывает метод своего делегата и отправляет уведомление **.textFieldDidBeginEditing(\_:)textDidBeginEditingNotification**
2. Текстовое поле вызывает различные методы делегата во время редактирования:

* Всякий раз, когда текущий текст изменяется, он вызывает метод и отправляет уведомление **.textField(\_:shouldChangeCharactersIn:replacementString:)** **.textDidChangeNotification**
* Он вызывает метод, когда пользователь нажимает встроенную кнопку, чтобы очистить текст **.textFieldShouldClear(\_:)**
* Он вызывает метод, когда пользователь нажимает кнопку возврата на клавиатуре **.textFieldShouldReturn(\_:)**

1. Перед тем, как покинуть пост первого респондента, текстовое поле вызывает метод своего делегата . Используйте этот метод для проверки текущего текста **.textFieldShouldEndEditing(\_:)**
2. Текстовое поле уходит с поста первого респондента.

В ответ система скрывает или настраивает клавиатуру по мере необходимости. Когда скрытие клавиатуры, системные сообщения на и уведомлений **.keyboardWillHideNotificationkeyboardDidHideNotification**

1. Текстовое поле вызывает метод своего делегата и отправляет уведомление **.textFieldDidEndEditing(\_:)textDidEndEditingNotification**

**Темы**

|  |  |
| --- | --- |
| **Управление редактированием** | func textFieldShouldBeginEditing(UITextField) -> Bool  Спрашивает делегата, начинать ли редактирование в указанном текстовом поле.  func textFieldDidBeginEditing(UITextField)  Сообщает делегату, когда начинается редактирование в указанном текстовом поле.  func textFieldShouldEndEditing(UITextField) -> Bool  Спрашивает делегата, прекратить ли редактирование в указанном текстовом поле.  func textFieldDidEndEditing(UITextField, reason: UITextField.DidEndEditingReason)  Сообщает делегату, когда редактирование останавливается для указанного текстового поля, и причину его остановки.  func textFieldDidEndEditing(UITextField)  Сообщает делегату об остановке редактирования для указанного текстового поля.  enum UITextField.DidEndEditingReason  Константы, указывающие причину завершения редактирования в текстовом поле. |
| **Редактирование текста текстового поля** | func textField(UITextField, shouldChangeCharactersIn: NSRange, replacementString: String) -> Bool  Спрашивает делегата, изменять ли указанный текст.  func textFieldShouldClear(UITextField) -> Bool  Спрашивает делегата, нужно ли удалить текущее содержимое текстового поля.  func textFieldShouldReturn(UITextField) -> Bool  Спрашивает делегата, обрабатывать ли нажатие кнопки возврата для текстового поля. |
| **Управление выделением текста** | func textFieldDidChangeSelection(UITextField)  Сообщает делегату об изменении выделения текста в указанном текстовом поле. |

**UITextViewDelegate**

Способы получения сообщений, связанных с редактированием, для объектов текстового представления.

**Декларация**

**@MainActor protocol UITextViewDelegate**

## **Обзор**

Все методы в этом протоколе не являются обязательными. Вы можете использовать их в ситуациях, когда вам может потребоваться отрегулировать текст, редактируемый пользователем (например, в случае программы проверки орфографии) или изменить предполагаемую точку вставки.

## **Темы**

|  |  |
| --- | --- |
| Ответ на уведомления об изменении | func textViewShouldBeginEditing(UITextView) -> Bool  Спрашивает делегата, начинать ли редактирование в указанном текстовом представлении.  func textViewDidBeginEditing(UITextView)  Сообщает делегату, когда начинается редактирование указанного текстового представления.  func textViewShouldEndEditing(UITextView) -> Bool  Спрашивает делегата, прекратить ли редактирование в указанном текстовом представлении.  func textViewDidEndEditing(UITextView)  Сообщает делегату об окончании редактирования указанного текстового представления. |
| Реагирование на изменения текста | func textView(UITextView, shouldChangeTextIn: NSRange, replacementText: String) -> Bool  Спрашивает делегата, заменить ли указанный текст в текстовом представлении.  func textViewDidChange(UITextView)  Сообщает делегату, когда пользователь изменяет текст или атрибуты в указанном текстовом представлении. |
| Реакция на изменения выбора | func textViewDidChangeSelection(UITextView)  Сообщает делегату об изменении выделения текста в указанном текстовом представлении. |
| Взаимодействие с текстовыми данными | func textView(UITextView, shouldInteractWith: NSTextAttachment, in: NSRange, interaction: UITextItemInteraction) -> Bool  Спрашивает делегата, разрешает ли указанное текстовое представление указанный тип взаимодействия пользователя с предоставленным текстовым вложением в указанном диапазоне текста.  func textView(UITextView, shouldInteractWith: URL, in: NSRange, interaction: UITextItemInteraction) -> Bool  Спрашивает делегата, разрешает ли указанное текстовое представление указанный тип взаимодействия пользователя с указанным URL-адресом в указанном диапазоне текста. |
| Константы | enum UITextItemInteraction  Константы, указывающие на тип взаимодействия, ожидаемого пользователем с URL-адресом или текстовым вложением. |