

Диалоговое окно ([англ. dialog box](#)) в графическом пользовательском интерфейсе — специальный [элемент интерфейса](#), окно, предназначенное для вывода информации и (или) получения ответа от пользователя. Получил своё название потому, что осуществляет двустороннее взаимодействие компьютер-пользователь («[диалог](#)»): сообщая пользователю что-то и ожидая от него ответа.

Диалоговые окна подразделяются на модальные и немодальные, в зависимости от того, блокируют ли они возможность взаимодействия пользователя с приложением (или системой в целом) до тех пор, пока не получат от него ответ.

Простейшим типом диалогового окна является окно сообщения ([англ. message box](#), [англ. alert box](#)), которое выводит сообщение и требует от пользователя подтвердить, что сообщение прочитано. Для этого обычно необходимо нажать [кнопку ОК](#). Окно сообщения предназначено для информирования пользователя о завершении выполнявшегося действия, вывода сообщения об ошибке и тому подобных случаев, не требующих от пользователя какого-либо выбора.

Несмотря на то, что использование окна сообщения является часто применяемым [шаблоном проектирования](#), оно критикуется экспертами по [юзабилити](#) как неэффективное решение для поставленной задачи (защита от ошибочных действий пользователя)^[1], вдобавок, при наличии лучших альтернатив

Кнопки

В диалоговых окнах многих [программ](#) присутствуют кнопки [ОК](#) и *Отмена* (Cancel), нажимая первую из которых, [пользователь](#) выражает своё согласие с тем, что в этот момент отображает диалоговое окно, и тем самым закрывает его, приводя в действие сделанные в нём изменения, а вторая — закрывает окно без применения. Существуют варианты с единственной кнопкой [ОК](#) — в информирующих окнах, не подразумевающих каких-либо изменений, и с кнопками [ОК](#), *Применить* (Apply) и *Закреть* (Close), в окнах, результаты изменений в которых пользователь, по мнению автора программы, может оценить, не закрывая окна, и в случае неудовлетворительного результата продолжить внесение изменений.

Также хорошим тоном считается обеспечить в диалоговых окнах справочную систему — обычно это реализуется либо в виде кнопки *Справка* (Help), открывающей страницу документации, описывающую работу в этом диалоговом окне, либо (в Windows) с помощью кнопки со знаком вопрос в заголовке окна, нажав на которую, пользователь переключает мышь в режим одноразовой контекстной подсказки: нажатие мышью в этом режиме на элемент диалогового окна вызывает отображение описания этого элемента.

Немодальные диалоговые окна

Немодальные ([англ. modeless](#)) диалоговые окна используются в случаях, когда выводимая в окне информация не является существенной для дальнейшей работы системы. Поэтому окно может оставаться открытым, в то время как работа пользователя с системой продолжается. Разновидностью немодального окна является [панель инструментов](#) или окно-«палитра», если она отсоединена или может быть отсоединена от главного окна приложения, так как элементы управления, расположенные на ней, могут использоваться параллельно с работой приложения. Впрочем, такие окна редко называют «диалоговыми».

В общем случае, правила хорошего [программного дизайна](#) предлагают использовать именно этот тип диалоговых окон, так как он не принуждает пользователя к определённой (возможно, не очевидной для него) последовательности действий. Примером может быть диалоговое окно для изменения каких-либо свойств текущего документа, например, цвета фона или текста. Пользователь может вводить текст, не обращая внимания на его цвет. Но в любой момент может изменить цвет, используя диалоговое окно. (*Приведённый пример не обязательно является лучшим решением. Такую же функциональность может обеспечить и панель инструментов, расположенная в главном окне.*)

Модальные на уровне приложения

Модальным называется окно, которое блокирует работу пользователя с родительским приложением до тех пор, пока пользователь это окно не закроет. Диалоговые окна преимущественно реализованы модальными.

Например, модальными являются диалоговые окна настроек приложения — так как проще реализовать режим, когда все сделанные изменения настроек применяются или отменяются одномоментно, и с момента, когда пользователь решил изменить настройки приложения и открыл диалог настроек, и до момента, когда он новые настройки вводит в силу или отказывается от них, приложение ожидает решения пользователя.

Отображение окон в модальном режиме практикуется и в других случаях, когда приложению для продолжения начатой работы требуется дополнительная информация, либо просто подтверждение от пользователя на согласие выполнить запрошенную последовательность действий, если она потенциально опасна.

Специалисты по [юзабилити](#) считают модальные окна запросов подтверждений плохими дизайнерскими решениями, так как они могут приводить пользователя к, так называемым, «[ошибкам режима программы](#)». Опасные действия должны иметь возможность отмены везде, где это возможно, а модальные окна запросов подтверждений, пугающие пользователя своим неожиданным появлением, через некоторое время пропускаются им автоматически (так как он привык к ним) и, поэтому, не защищают от опасных действий.

Модальные на уровне окна

Для смягчения недостатков модальных диалоговых окон (блокирование доступа к приложению, или, даже, к системе в целом) была предложена концепция модальности на уровне окна (или документа). Такой тип диалогового окна введён, например, в [Mac OS X](#), где он называется «диалоговое модальное окно для документа» ([англ. document modal sheet](#)).

При появлении диалогового окна подобного типа работа с остальными окнами приложения не блокируется. Поскольку окно диалога не перекрывает родительское окно, а прикрепляется к нему, остаётся возможность изменять размеры и положение и самого родительского окна. При этом, естественно, содержимое родительского окна остаётся недоступным до закрытия модального диалогового окна.

Диалоговые окна

Пакет `tkinter` содержит несколько модулей, предоставляющих доступ к уже готовым диалоговым окнам. Это окна различных сообщений, выбора по принципу "да-нет", открытия и сохранения файлов и др. В этом уроке рассмотрим примеры окон из модулей `messagebox` и `filedialog` пакета `tkinter`.

Модули пакета необходимо импортировать отдельно. То есть вы импортируете содержимое `tkinter` (например, `from tkinter import *`) и отдельно входящий в состав пакета `tkinter` модуль. Способы импорта на примере `messagebox` и пример вызова одной из функций модуля:

- `import tkinter.messagebox → tkinter.messagebox.askyesno()`
- `from tkinter.messagebox import * → askyesno()`
- `from tkinter import messagebox → messagebox.askyesno()`
- `from tkinter import messagebox as mb (вместо mb может быть любой идентификатор) → mb.askyesno()`

В уроке мы будем использовать последний вариант.

Модуль `messagebox` – стандартные диалоговые окна

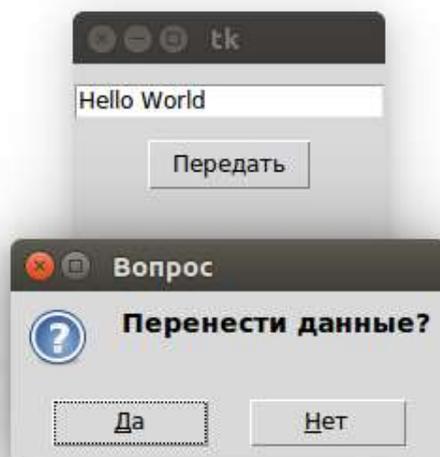
Окно выбора "да" или "нет" – `askyesno()`:

```
from tkinter import *
from tkinter import messagebox as mb

def check():
    answer = mb.askyesno(title="Вопрос", message="Перенести данные?")
    if answer == True:
        s = entry.get()
        entry.delete(0, END)
        label['text'] = s

root = Tk()
entry = Entry()
entry.pack(pady=10)
Button(text='Передать', command=check).pack()
label = Label(height=3)
label.pack()

root.mainloop()
```



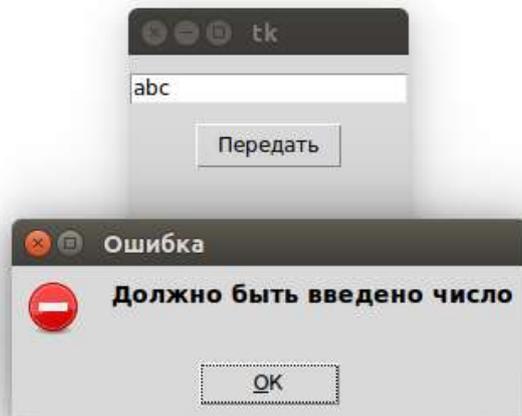
Нажатие "Да" в диалоговом окне возвращает в программу True, "Нет" вернет False (также как закрытие окна через крестик). Таким образом в коде можно обработать выбор пользователя. В данном случае если последний соглашается, то данные переносятся из поля в метку.

Опции `title` и `message` являются позиционными, так что можно указывать только значения: `askyesno("Вопрос", "Перенести данные?")`.

Подобные окна генерируются при использовании функции `askokcancel()` с надписями на кнопках "ОК" и "Отмена", `askquestion()` (возвращает не True или False, а строки 'yes' или 'no'), `askretrycancel()` ("Повторить", "Отмена"), `askyesnocancel()` ("Да", "Нет", "Отмена"). Другую группу составляют окна с одной кнопкой, которые служат для вывода сообщений различного характера. Это `showerror()`, `showinfo()` и `showwarning()`.

```
...
def check():
    s = entry.get()
    if s.isdigit() == False:
        mb.showerror("Ошибка", "Должно быть введено число")
    else:
        entry.delete(0, END)
        label['text'] = s
...

```



Модуль filedialog – диалоговые окна открытия и сохранения файлов

Рассмотрим две функции из модуля filedialog – askopenfilename() и asksaveasfilename(). Первая предоставляет диалоговое окно для открытия файла, вторая – для сохранения. Обе возвращают имя файла, который должен быть открыт или сохранен, но сами они его не открывают и не сохраняют. Делать это уже надо программными средствами самого Python.

```
from tkinter import *
from tkinter import filedialog as fd

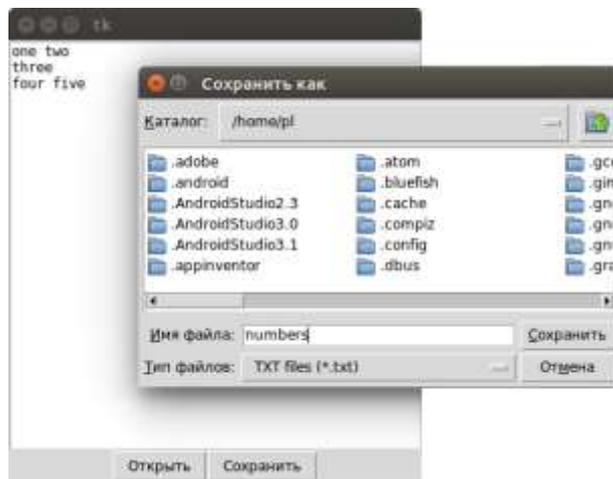
def insertText():
    file_name = fd.askopenfilename()
    f = open(file_name)
    s = f.read()
    text.insert(1.0, s)
    f.close()

def extractText():
    file_name = fd.asksaveasfilename(filetypes=(("TXT files", "*.txt"),
                                                ("HTML files", "*.html;*.htm"),
                                                ("All files", "*.*") ))

    f = open(file_name, 'w')
    s = text.get(1.0, END)
    f.write(s)
    f.close()

root = Tk()
text = Text(width=50, height=25)
text.grid(columnspan=2)
b1 = Button(text="Открыть", command=insertText)
b1.grid(row=1, sticky=E)
b2 = Button(text="Сохранить", command=extractText)
b2.grid(row=1, column=1, sticky=W)

root.mainloop()
```



Опция `filetype` позволяет перечислить типы файлов, которые будут сохраняться или открываться, и их расширения.

Примечание. В приведенном коде при размещении текстового поля методом `grid()` не указаны аргументы `row` и `column`. В таких случаях подразумевается, что их значениями являются нули.