# 6.2. Листинги

|  |
| --- |
| Облачное хранение файлов на file.io  Заготовка проекта  from tkinter import \*  import requests  from tkinter import filedialog as fd  from tkinter import ttk  def upload():  filepath = fd.askopenfilename()  if filepath:  files = {'file': open(filepath, 'rb')}  response = requests.post('https://file.io', files=files)  if response.status\_code == 200:  download\_link = response.json()['link']  link\_entry.insert(0, download\_link)  window = Tk()  window.title("Сохранение файлов в облаке")  window.geometry(“400x200”)  upload\_button = ttk.Button(text="Загрузить файл", command=upload)  upload\_button.pack()  entry = ttk.Entry()  entry.pack()  window.mainloop() |
|  |
| Добавляем обработку исключений  from tkinter import \*  import requests  from tkinter import filedialog as fd  from tkinter import messagebox as mb  from tkinter import ttk  def upload():  try:  filepath = fd.askopenfilename()  if filepath:  with open(filepath, 'rb') as f:  files = {'file': f}  response = requests.post('https://file.io', files=files)  response.raise\_for\_status() # Проверка на ошибки HTTP  download\_link = response.json().get('link')  if download\_link:  entry.delete(0, END)  entry.insert(0, download\_link)  else:  raise ValueError("Не удалось получить ссылку для скачивания")  except ValueError as ve:  mb.showerror("Ошибка", f"Произошла ошибка: {ve}")  except Exception as e:  mb.showerror("Ошибка", f"Произошла ошибка: {e}")  window = Tk()  window.title("Сохранение файлов в облаке")  window.geometry("360x100")  upload\_button = ttk.Button(text="Загрузить файл", command=upload)  upload\_button.pack()  entry = ttk.Entry()  entry.pack()  window.mainloop() |
| Автоматическое копирование ссылки в буфер обмена. Сначала установим pyperclip  import requests  import pyperclip  from tkinter import filedialog, messagebox  from tkinter import ttk  import tkinter as tk  def upload\_file():  try:  filepath = filedialog.askopenfilename()  if filepath:  with open(filepath, 'rb') as f:  files = {'file': f}  response = requests.post('https://file.io', files=files)  response.raise\_for\_status() # Проверка на ошибки HTTP  download\_link = response.json().get('link')  if download\_link:  link\_entry.delete(0, tk.END)  link\_entry.insert(0, download\_link)  pyperclip.copy(download\_link) # Копирование ссылки в буфер обмена  messagebox.showinfo("Ссылка скопирована", "Ссылка успешно скопирована в буфер обмена")  else:  raise ValueError("Не удалось получить ссылку для скачивания")  except Exception as ex:  messagebox.showerror("Ошибка", f"Произошла неизвестная ошибка: {ex}")  app = tk.Tk()  app.title("TempFile Share")  upload\_button = ttk.Button(app, text="Upload File", command=upload\_file)  upload\_button.pack()  link\_entry = ttk.Entry(app)  link\_entry.pack()  app.mainloop() |
| Сохранение в формате JSON  import json  import os  import requests  import pyperclip  from tkinter import filedialog, messagebox, Toplevel, Listbox, Scrollbar, HORIZONTAL, VERTICAL  from tkinter import ttk  import tkinter as tk  history\_file = "upload\_history.json"  def save\_history(file\_path, download\_link):  history = []  if os.path.exists(history\_file):  with open(history\_file, "r") as file:  history = json.load(file)    history.append({"file\_path": os.path.basename(file\_path), "download\_link": download\_link})    with open(history\_file, "w") as file:  json.dump(history, file, indent=4)  def upload\_file():  try:  filepath = filedialog.askopenfilename()  if filepath:  with open(filepath, 'rb') as f:  files = {'file': f}  response = requests.post('https://file.io', files=files)  response.raise\_for\_status()  download\_link = response.json().get('link')  if download\_link:  link\_entry.delete(0, tk.END)  link\_entry.insert(0, download\_link)  pyperclip.copy(download\_link)  save\_history(filepath, download\_link)  messagebox.showinfo("Ссылка скопирована", "Ссылка успешно скопирована в буфер обмена")  else:  raise ValueError("Не удалось получить ссылку для скачивания")  except requests.RequestException as e:  messagebox.showerror("Ошибка сети", f"Произошла ошибка сети: {e}")  except ValueError as ve:  messagebox.showerror("Ошибка", str(ve))  except Exception as ex:  messagebox.showerror("Ошибка", f"Произошла неизвестная ошибка: {ex}")  app = tk.Tk()  app.title("TempFile Share")  upload\_button = ttk.Button(app, text="Upload File", command=upload\_file)  upload\_button.pack()  link\_entry = ttk.Entry(app)  link\_entry.pack()  app.mainloop() |
| Точки остановки (breakpoints) используются для отладки программы, вот несколько ключевых мест, где может быть полезно установить точки остановки для отладки:  4. \*\*Перед Сохранением Истории в JSON:\*\*  - В функции `save\_history`, перед записью данных в JSON-файл (`json.dump(...)`).  - Это даст возможность убедиться, что данные для сохранения сформированы корректно и файл открывается для записи без ошибок.  5. \*\*В Начале Функции Показа Истории:\*\*  - В функции `show\_history`, в самом начале.  - Здесь можно проверить, успешно ли открывается и читается JSON-файл, и корректно ли извлекаются данные истории.  6. \*\*При Обработке Исключений:\*\*  - В блоках `except` в функции `upload\_file`.  - Полезно для понимания, какие исключения возникают и как они обрабатываются.  Установка точек остановки на этих этапах поможет вам тщательно отладить ключевые аспекты программы, включая взаимодействие с внешним API, обработку файлов, работу с JSON и обработку исключений. |
| Загрузка JSON  import json  import os  import requests  import pyperclip  from tkinter import \*  from tkinter import ttk  import tkinter as tk  history\_file = "upload\_history.json"  def save\_history(file\_path, download\_link):  history = []  if os.path.exists(history\_file):  with open(history\_file, "r") as file:  history = json.load(file)    history.append({"file\_path": os.path.basename(file\_path), "download\_link": download\_link})    with open(history\_file, "w") as file:  json.dump(history, file, indent=4)  def show\_history():  if not os.path.exists(history\_file):  messagebox.showinfo("История", "История загрузок пуста")  return  history\_window = Toplevel(window)  history\_window.title("История Загрузок")  files\_listbox = Listbox(history\_window, width=50, height=20)  files\_listbox.grid(row=0, column=0, padx=(10,0), pady=10)  links\_listbox = Listbox(history\_window, width=50, height=20)  links\_listbox.grid(row=0, column=1, padx=(0,10), pady=10)  with open(history\_file, "r") as file:  history = json.load(file)  for item in history:  files\_listbox.insert(END, item['file\_path'])  links\_listbox.insert(END, item['download\_link'])  def upload\_file():  # ... остальной код функции upload\_file ...  app = tk.Tk()  app.title("TempFile Share")  # ... остальной код для создания главного окна …  history\_button = ttk.Button(app, text="Показать Историю", command=show\_history)  history\_button.pack()  app.mainloop() |
| Юнит тест для функции save\_history  import json  import os  history\_file = "test\_upload\_history.json"  def save\_history(file\_path, link):  history = []  if os.path.exists(history\_file):  with open(history\_file, 'r') as f:  history = json.load(f)  history.append({"file\_path": os.path.basename(file\_path), "download\_link": link})  with open(history\_file, 'w') as f:  json.dump(history, f, indent=4)  def test\_save\_history():  test\_file\_path = "test\_file.txt"  test\_download\_link = "https://file.io/example"  # Вызов функции для тестирования  save\_history(test\_file\_path, test\_download\_link)  # Проверка, что история была сохранена корректно  with open("test\_upload\_history.json", "r") as file:  history = json.load(file)  assert len(history) == 1  assert history[0]['file\_path'] == test\_file\_path  assert history[0]['download\_link'] == test\_download\_link  # Очистка тестовых данных  os.remove("test\_upload\_history.json")  # Вызов функции тестирования  test\_save\_history() |