### **Kodify Nano**

**Kodify Nano** – ИИ-ассистент для программистов, помогающий в написании программного кода и решении других задач разработки, таких как рефакторинг, тестирование и документирование. Ассистент поддерживает основные языки программирования, включая Python, Java, JavaScript, C# и Go, и может переводить код с одного языка на другой.

Kodify Nano представляет собой LLM и плагин, встраиваемый в IDE от JetBrains (например, PyCharm) и Visual Studio Code. Установка плагина в IDE даёт разработчику возможность обращаться к LLM Kodify Nano для автоматической генерации кода на основе данных из редактируемого пользователем файла, а также для решения других задач. LLM модель Kodify Nano содержит 1.5 миллиарда параметров и может быть запущена на видеокартах от 4 Гб видеопамяти. Модель разработана для выполнения задач разработки кода с минимальными ресурсами. Она оптимизирована для быстрого и эффективного взаимодействия с пользователями, обеспечивая высокую производительность даже в условиях ограниченных ресурсов.

#### Использование образа

Вы можете запустить Kodify Nano на OLLAMA двумя способами:

- используя Docker;
- локально. При этом способе запуска модель отвечает быстрее, чем при запуске через Docker.

#### Способ 1. Запуск Kodify Nano на OLLAMA в Docker

1. Если у вас нет видеокарты NVIDIA, запустите OLLAMA следующей командой:

docker run -e OLLAMA\_HOST=0.0.0.0:8985 -p 8985:8985 --name ollama -d ollama/ollama

Убедитесь, что у вас установлен и запущен Docker перед запуском команды.

Важно! Если порт 8985 уже занят, замените его на любой другой свободный порт. В этом случае также требуется изменить порт в конфигурации плагина. Смотрите раздел "Изменение адреса порта в настройках плагина в средах Visual Studio Code и JetBrains".

Если у вас есть видеокарта NVIDIA и нужно добавить GPU в контейнер, выполните следующую команду:

docker run --runtime nvidia -e OLLAMA\_HOST=0.0.0.0:8985 -p 8985:8985 --name ollama -d ollama/ollama

2. Загрузите модель в OLLAMA:

docker exec ollama ollama pull hf.co/MTSAIR/Kodify-Nano-GGUF

где "hf.co/MTSAIR/Kodify-Nano-GGUF" - имя модели. Подробнее о модели смотрите в <u>https://huggingface.co/MTSAIR/Kodify-Nano-GGUF</u>.

3. Задайте корректное имя для модели.

Измените "hf.co/MTSAIR/Kodify-Nano-GGUF" на "kodify\_nano" следующей командой:

docker exec ollama ollama cp hf.co/MTSAIR/Kodify-Nano-GGUF kodify\_nano

4. Запустите модель:

docker exec ollama ollama run kodify\_nano

### Способ 2. Запуск Kodify Nano на OLLAMA локально

1. Скачайте OLLAMA с сайта командой:

https://ollama.com/download

2. Задайте порт:

export OLLAMA\_HOST=0.0.0.0:8985

Важно! Если порт 8985 уже занят, замените его на любой другой свободный порт. В этом случае также требуется изменить порт в конфигурации плагина. Смотрите раздел "Изменение адреса порта в настройках плагина в средах Visual Studio Code и JetBrains".

3. Поднимите сервер OLLAMA.

ollama serve &

4. Загрузите модель, выполнив следующую команду:

ollama pull hf.co/MTSAIR/Kodify-Nano-GGUF

где "hf.co/MTSAIR/Kodify-Nano-GGUF" - имя модели. Подробнее о модели смотрите в <u>https://huggingface.co/MTSAIR/Kodify-Nano-GGUF</u>.

- 5. Задайте корректное имя для модели. Измените hf.co/MTSAIR/Kodify-Nano-GGUF на kodify\_nano следующей командой:
   ollama cp hf.co/MTSAIR/Kodify-Nano-GGUF kodify\_nano
- 6. Запустите модель следующей командой:

ollama run kodify\_nano

Список других моделей Kodify Nano:

- <u>https://huggingface.co/MTSAIR/Kodify-Nano</u>
- <u>https://huggingface.co/MTSAIR/Kodify-Nano-CPU</u>
- <u>https://huggingface.co/MTSAIR/Kodify-Nano-GPTQ</u>

### Установка плагина

## Установка плагина в среде Visual Studio Code

- 1. Скачайте последнюю версию плагина Kodify для Visual Studio Code. Ссылку для скачивания вам предоставят.
- 2. Перейдите в раздел "Extensions" на левой панели инструментов.



3. В диалоговом меню выберите пункт "Install from VSIX..." и укажите скачанный файл плагина.



### Установка плагина в среде JetBrains

- 1. Скачайте последнюю версию плагина Kodify для JetBrains. Ссылку для скачивания вам предоставят.
- 2. Запустите IDE и перейдите в настройки IDE.
- 3. Выберите пункт "Plugins" для перехода в настройку плагинов.

Pc PyCharm 2022.3.2	Marketplace		
	Q• Type / to see options		
Projects Customize	Settings Sync  bundled		
Plugins	Languages	Disable all	
Learn	Markdown  bundled	<	

4. Нажмите на шестеренку в правом верхнем углу и выберите "Install Plugin from Disk...".

		Preferences				
Q <sup>r</sup> plugins ×	Plugins	Marketplace	Installed 🌣			
Plugins       Image: Comparison of the second			110 H	anage Plugin Repositories TTP Proxy Settings	positories	
	BinEd - Binary/Hex Ed	litor 🗸	Manage Plugin Contificated Install Plugin from Disk Disable All Downloaded Plugins Enable All Downloaded Plugins Binary/hex concorprogram cooce on once more more plugins • Use "Open as Binary" action in toolbar of		"File/Open" dialo	
	Gradle 222.4459.17 JetBrains					
	M T MTS Copilot C 1.2.0-dev MTS		menu • Use "View a • Associate fi	u in variables/d type in Option		
	PlantUML Integration     6.5.0-IJ2022.2 Eugene St		• Use "Byte- • Use "Edit a	<ul> <li>Use "Byte-to-byte compare" in Compare files d</li> <li>Use "Edit as Binary" in column context menu in</li> </ul>		
	req.txt Requirements 2022.4.1 meanmail		Preview			
	Run Configuration for 2022.11.14-1 bluelovers	TypeScript 🛛 🗸				
	Vue.js 222.4459.28 JetBrains		Features			
			<ul> <li>Visualize da</li> <li>Codes can</li> </ul>	ata as numerical (hexadecimal) c be also binary, octal or decimal	odes and text	
	Database Tools and S bundled	QL 🗹	<ul> <li>Support for</li> <li>Insert and o</li> <li>Searching f</li> <li>Support for</li> </ul>	<ul> <li>Support for Unicode, UTF-8 and other charsets</li> <li>Insert and overwrite edit modes</li> <li>Searching for text / hexadecimal code with found mails support for undo/redo</li> </ul>		
				Cancel Apply	ОК	

- 5. Выберите скачанный файл с плагином.
- 6. Подтвердите перезагрузку IDE после установки. Нажмите "Restart IDE".

# Изменение адреса порта в настройках плагина в средах Visual Studio Code и JetBrains

Если порт при поднятии Docker Compose отличается от порта **8985**, измените порт в файле *config.json*, выполнив следующие шаги:

- 1. Откройте в IDE любой файл.
- 2. Откройте боковую панель Kodify, используя следующую комбинацию клавиш:
  - в Visual Studio Code "CTRL + L" ("CMD + L" на Mac);



• в JetBrains - "CTRL+ J" ("CMD + J" на Mac).



3. Откройте конфигурационный файл *config.json* одним из следующих способов:

### Способ 1

На боковой панели Kodify нажмите на кнопку "Open Settings" в Visual Studio Code



или "Kodify Config" в JetBrains.



В разделе "Configuration" для Chat настроек нажмите кнопку "Open Config File".



## Способ 2

На боковой панели раскройте меню Kodify и нажмите на кнопку "Open Settings" (шестеренка).



- 4. Измените в конфигурационном файле *config.json* адрес порта для поля "apiBase" в разделах конфигурации плагина "tabAutocompleteModel" и "models".
- 5. Сохраните настройки файла *config.json* в IDE. Для этого, нажмите комбинацию клавиш **Ctrl+S** или выберите пункт меню IDE: **File -> Save**.