

```
        _replace('/\\\\\\\' , '/', $image_src)
    $A['config'] = serialize($captcha_config);
    $tcha_config['code'],
    $image_src
}

<2rgb') ) {
    $str, $return_string = false, $separator = ',')
    _replace("/[^0-9A-Fa-f]/", '', $hex_str); // Gets
    /();
    r) == 6 ) {
        exdec($hex_str);
    ] = 0xFF & ($color_val >> 0x10);
    = 0xFF & ($color_val >> 0x8);
    = 0xFF & $color_val;
    ex_str) == 3 ) {
        = hexdec(str_repeat(substr($hex_str, 0, 1), 2));
        = hexdec(str_repeat(substr($hex_str, 1, 1), 2));
        = hexdec(str_repeat(substr($hex_str, 2, 1), 2));
    ? implode($separat
```

# Prettier

Автоматичне форматування коду для сучасних розробників

# and organized code

```
import passport from 'passport';
import LocalStrategy from 'passport-local';
import { Strategy as JWTStrategy, ExtractJwt } from 'passport-jwt';
import request from 'superagent';

import User from '../models/user.model';
import constants from '../config/constants';
import { createUser } from '../helpers/auth.helper';

/**
 * Local Strategy Auth
 */
const localOpts = { usernameField: 'username' };
```

## Що таке Prettier?

Prettier - це інструмент для форматування коду, який автоматично налаштовує форматування вашого коду відповідно до встановлених правил. Він працює як лінтер, але зосереджений саме на форматуванні коду, а не на виявленні помилок.

На відміну від традиційних лінтерів, Prettier не просто вказує на проблеми - він активно переписує ваш код, забезпечуючи консистентний стиль у всьому проекті. Це означає, що ви більше не витрачаєте час на суперечки про те, де ставити фігурні дужки або скільки пробілів використовувати для відступів.

- ❑💡 Prettier називають "opinionated formatter" – він приймає рішення за вас, залишаючи мінімум налаштувань і максимум продуктивності.

# Основні особливості Prettier



## Автоматичне форматування

Prettier переписує ваш код, дотримуючись встановлених стандартів форматування без вашого втручання



## Конфігурація

Налаштуйте правила форматування: відступи, вирівнювання, лапки та багато іншого



## Підтримка багатьох мов

JavaScript, TypeScript, CSS, HTML, JSON, Markdown та інші популярні мови



## Інтеграція з редакторами

Плагіни для VS Code, WebStorm, Sublime Text, Atom та інших IDE



## Командний рядок

Форматуйте файли та цілі проекти через CLI для автоматизації процесів



## Командна робота

Забезпечує єдиний стиль коду в проекті, полегшуєчи читання для всієї команди

Використання Prettier спрощує життя розробників, усуваючи суперечки про стиль коду та дозволяючи зосередитися на логіці програми.

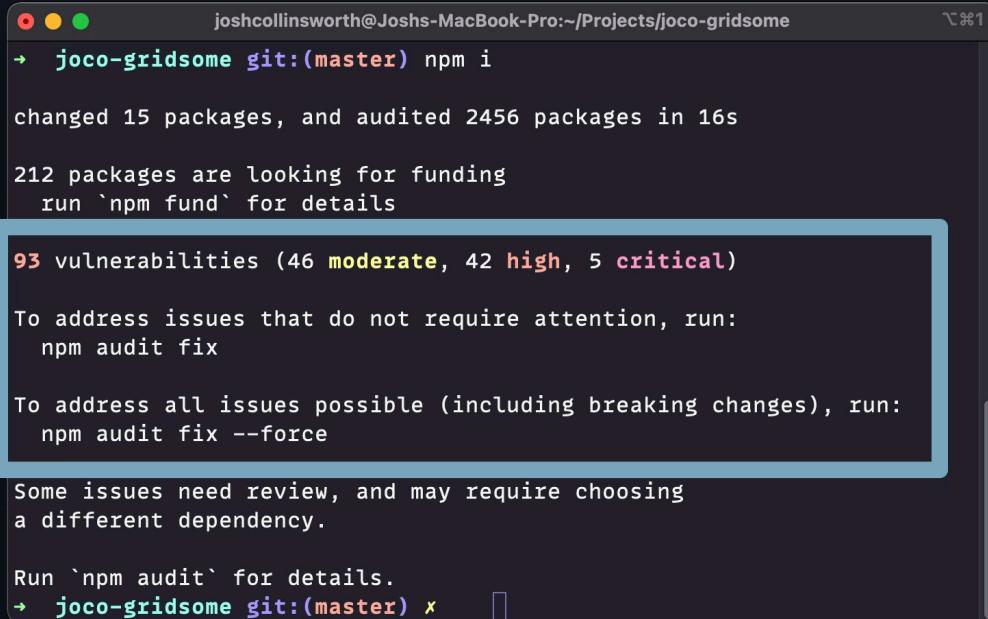
# Інсталяція та початкова конфігурація

## Встановлення Prettier

Для локального встановлення у вашому проекті виконайте команду:

```
npm install --save-dev --save-exact prettier
```

Прапорець `--save-exact` гарантує, що всі члени команди використовують однакову версію.



The terminal window shows the following output:

```
joshcollinsworth@Joshs-MacBook-Pro:~/Projects/joco-gridsome
→ joco-gridsome git:(master) npm i
changed 15 packages, and audited 2456 packages in 16s
212 packages are looking for funding
  run `npm fund` for details

93 vulnerabilities (46 moderate, 42 high, 5 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues possible (including breaking changes), run:
  npm audit fix --force

Some issues need review, and may require choosing
a different dependency.

Run `npm audit` for details.
→ joco-gridsome git:(master) x
```

## Створення конфігураційного файлу

Конфіг файл визначає правила форматування для вашого коду. Prettier підтримує різні формати:

### .prettierrc

JSON або YAML формат - найпопулярніший варіант

### .prettierrc.json

Явний JSON формат для максимальної чіткості

### .prettierrc.js

JavaScript модуль з `export default` або `module.exports`

### prettier.config.js

Альтернативна назва для JS конфігурації

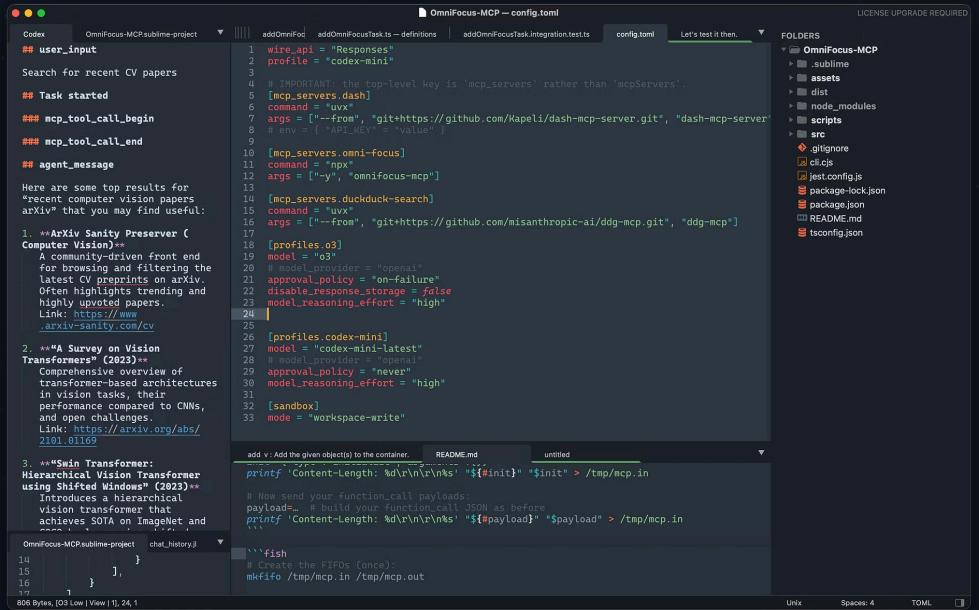
# Налаштування проекту

## Створення конфігураційного файлу

Створіть файл `.prettierrc` за допомогою команди:

```
node --eval "fs.writeFileSync('.prettierrc','{}\\n')"
```

Або створіть його вручну у кореневій директорії проекту.



```
user_input
Search for recent CV papers
## Task started
### mcp_tool_call_begin
### mcp_tool_call_end
## agent_message
Here are some top results for
"recent computer vision papers
arXiv"
that you may find useful:
1. **Arxiv Sanity Preserver (Computer Vision)**
A community-driven front end
for browsing and filtering the
latest CV preprints on Arxiv.
Often highlights trending and
highly-cited papers.
Link: https://www.arxiv-sanity.com/cv
2. **A Survey on Vision Transformers** (6823)
Comprehensive overview of
transformer-based architectures
in vision tasks, their
performance compared to CNNs,
and open challenges.
Link: https://arxiv.org/abs/2101.01169
3. **Swin Transformer: Hierarchical Vision Transformer using Multi-Scale Patch Merging (6929)**
Introduces a hierarchical
vision transformer that
achieves SOTA on ImageNet and
other benchmarks.
Link: https://arxiv.org/abs/2103.14030
## fish
Create the FIFOs (once):
mkfifo /tmp/mcp_in /tmp/mcp.out
```

## Файл `.prettierignore`

Створіть `.prettierignore` для виключення файлів і папок з форматування:

```
# Ignore artifacts:
build
coverage

# Dependencies
node_modules/

# Rest
test.js
*.min.js
```

- Синтаксис `.prettierignore` ідентичний до `.gitignore` - використовуйте глоби для гнучкого керування виключеннями.

# Правила форматування

Розглянемо приклад конфігурації з основними правилами:

```
{  
  "singleQuote": true,  
  "semi": true,  
  "bracketSpacing": true,  
  "useTabs": true,  
  "printWidth": 120  
}
```



## singleQuote

Використовувати одинарні лапки замість подвійних



## semi

Додавати крапку з комою в кінці кожного рядка



## bracketSpacing

Пробіли між фігурними дужками в об'єктах



## useTabs

Використовувати табуляцію для відступів



## printWidth

Максимальна довжина рядка - 120 символів

Після застосування цих правил ваш код буде автоматично відформатований відповідно до встановлених стандартів. Повний список доступних правил та їх детальний опис можна знайти в офіційній документації Prettier.



## Запуск з командного рядка

Prettier надає потужний CLI для форматування коду безпосередньо з терміналу. Це особливо корисно для автоматизації та інтеграції в процеси CI/CD.

1

### Базове форматування

```
npx prettier . --write
```

Форматує всі файли в поточній директорії

2

### Перевірка без змін

```
npx prettier . --check
```

Перевіряє форматування без модифікації файлів

3

### Конкретні файли

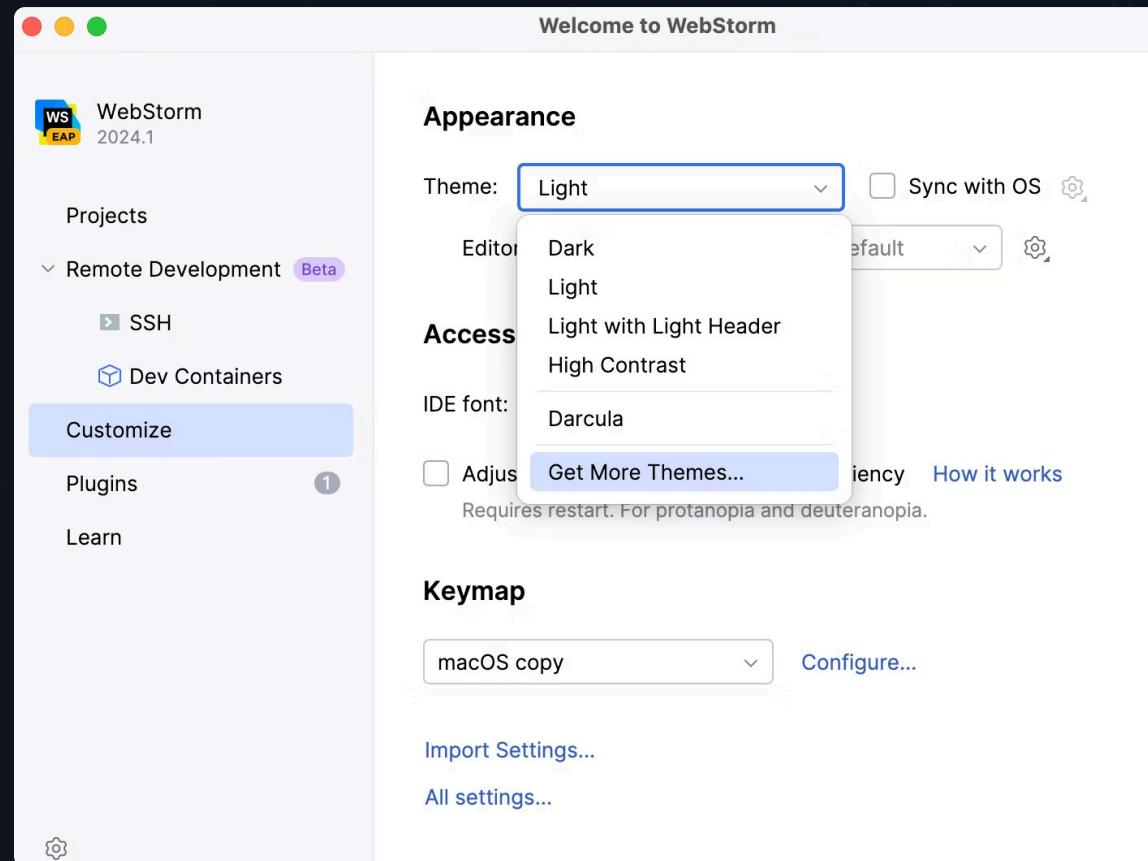
```
npx prettier src/**/*.{js} --write
```

Форматує тільки JavaScript файли в папці src

- Корисна порада:** Додайте скрипт до package.json для швидкого доступу: "format": "prettier . --write"

Повний список опцій CLI включає налаштування для різних сценаріїв використання - від простого форматування до складної інтеграції з іншими інструментами. Детальну документацію можна знайти на офіційному сайті Prettier.

# Інтеграція з WebStorm / IntelliJ IDEA



## Вбудована підтримка

WebStorm має вбудовану підтримку Prettier. Для інших IDE JetBrains (IntelliJ IDEA, PhpStorm, PyCharm) переконайтесь, що плагін встановлено через **Settings / Preferences | Plugins**.

## Можливості форматування

### Ручне форматування

Використовуйте **Reformat with Prettier**:

- macOS: Opt+Shift+Cmd+P
- Windows/Linux: Alt+Shift+Ctrl+P

### Форматування при збереженні

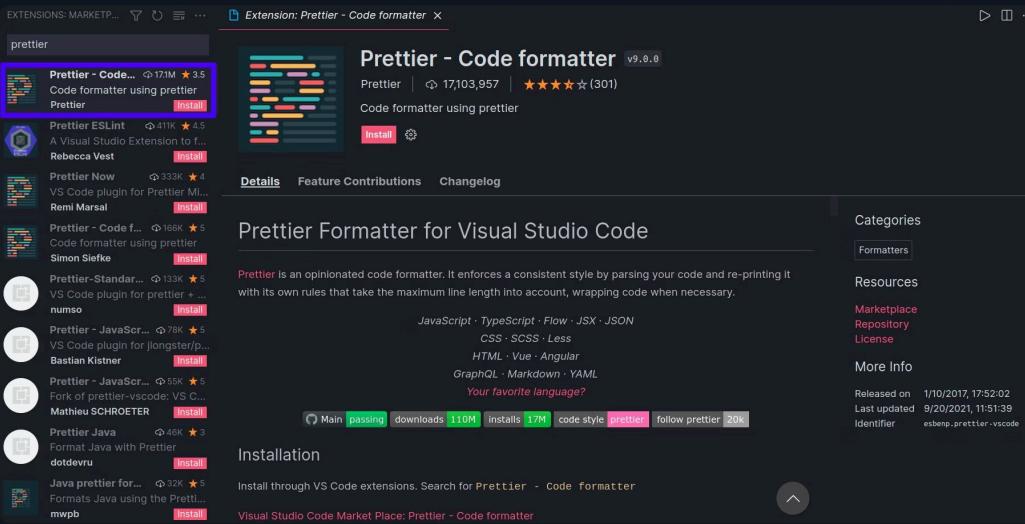
Автоматично запускає Prettier при натисканні Cmd+S або Ctrl+S. Активуйте в налаштуваннях: **Languages & Frameworks | JavaScript | Prettier**

### Форматування за замовчуванням

Використовуйте Prettier як основний форматер для Ctrl+Alt+L / Opt+Cmd+L

За замовчуванням WebStorm форматує файли .js, .ts, .jsx і .tsx. Налаштуйте глобальні шаблони для розширення підтримки інших типів файлів або обмеження форматування певними директоріями.

# Visual Studio Code



## Встановлення плагіну

Знайдіть та встановіть розширення "Prettier - Code formatter" через marketplace VS Code.

Після встановлення плагін автоматично інтегрується з редактором і надає широкі можливості для налаштування.

## Швидке налаштування

### 1 Встановіть Prettier як форматер за замовчуванням

У налаштуваннях VS Code встановіть "editor.defaultFormatter": "esbenp.prettier-vscode"

### 2 Увімкніть форматування при збереженні

Додайте "editor.formatOnSave": true для автоматичного форматування

### 3 Налаштуйте для конкретних мов

Використовуйте специфічні налаштування для різних типів файлів за потреби

□ Детальна документація та приклади конфігурації доступні на GitHub: <https://github.com/prettier/prettier-vscode>

Плагін prettier-vscode надає інтуїтивний інтерфейс для роботи з Prettier безпосередньо у вашому улюбленаому редакторі, роблячи процес форматування коду максимально зручним та ефективним.