```ts
import { Command, Option, Cli, Builtins, CliOptions } from 'clipanion'
import { Theme } from 'shiki'
import { bgGreen, black } from 'colorette'
import { compilePdfs } from '.'

const defineUsage = Command.Usage

export class RootCommand extends Command {
  static paths = [Command.Default]

  static usage = defineUsage({
    description: 'compile source code to pdfs',
    examples: [
      [
          'Compile all files in src (will write to the pdfs directory by default)',
        '$0 src pdfs',
      ],
      ['Compile all files in src to the dist directory', '$0 src -d dist'],
      ['Compile only javascript files', '$0 src --include "src/**/*.js"'],
      ['Exclude tests from compilation', '$0 src --exclude "src/**/*.test.js"'],
      ['Specify a theme', '$0 src -t github-light'],
    ],
  })

  rootDir = Option.String()
  outDir = Option.String('-d,--out-dir')
  include = Option.Array('--include')
  exclude = Option.Array('--exclude')

  theme: Theme | undefined = Option.String('-t,--theme')

  async execute() {
    const include = this.include ?? [this.rootDir]

    const { count: compiledFilesCount } = await compilePdfs({
      rootDir: this.rootDir,
      outDir: this.outDir,
      include,
      exclude: this.exclude,
      theme: this.theme,
    })

    const label = bgGreen(black(' PDF '))

    this.context.stdout.write(
      `${label} Successfully compiled ${compiledFilesCount} files\n`
    )
  }
}

export const getCli = (options: Partial<CliOptions>) => {
  const cli = new Cli({
    binaryLabel: 'pdfc',
    ...options,
  })

  cli.register(Builtins.HelpCommand)
  cli.register(Builtins.VersionCommand)
  cli.register(RootCommand)

  return cli
}
```