The code-first reel starter

From the reel "Claude made this reel in code, not in a video app"

From @learningwithcole

Every reel on this page is a React file, not a project in a video editor. Remotion turns code into an MP4, and Claude Code writes the code. Here's the smallest setup that works.

1. Set it up (one command)

You need Node.js. Then:

Terminal

```
npx create-video@latest
```

Pick a blank template, open the folder in your terminal, and start Claude Code there.

2. One composition

A composition is a React component plus the video's size, length and frame rate. You register it in `src/Root.tsx`:

TypeScript

```
<Composition
  id="MyReel"
  component={MyReel}
  durationInFrames={30 * 40} // 40 s at 30 fps
  fps={30}
  width={1080}
  height={1920}
/>
```

Inside `MyReel`, `useCurrentFrame()` tells you which frame you're drawing. A video is just what your component returns for frame 0, 1, 2 and so on. Animation = a value that depends on the frame.

Prompt to give Claude:

Prompt

Build a 1080×1920 Remotion composition called MyReel. Scene 1 shows the headline "[your hook]" from frame 0. Scene 2 shows [thing]. Scene 3 shows [thing]. Keep every scene's start frame in one object at the top of the file so I can retime it.

3. Time every scene to your voice

Put your voiceover in the `public/` folder and add it to the composition:

TypeScript

```
import { staticFile } from "remotion";
import { Audio } from "@remotion/media";

<Audio src={staticFile("vo.wav")} />
```

1. Get word timestamps for the voiceover (start and end in seconds for each word) from any speech-to-text tool that outputs them, and save them as `words.json`.
2. Turn a spoken word into a frame: `Math.round(word.start * fps)`.
3. Start each scene on a word, not on a hand-typed frame number. Write a tiny helper, `cue("here is the trick")`, that finds that phrase in `words.json` and returns its frame. Have it throw an error if the phrase isn't found, so a changed script can't silently break your timing.

Now if you re-record the voice faster or slower, every scene moves with it.

4. Preview and render

Terminal

```
npx remotion studio          # live preview in your browser
npx remotion render src/index.ts MyReel out/reel.mp4
```

The render runs on your own computer and writes an H.264 MP4. No upload, no video model.

5. Change one line, render again

Keep the things you tweak most at the top of the file:

TypeScript

```typescript
const ACCENT = "#E8553D";
const HOOK = "Claude made this reel in code";
```

Change one value, run the render command again, and every scene that uses it updates. Same goes for the ending: it's a component, so swap it and re-render.

Good to know

- Remotion is free for individuals and for companies with up to 3 employees. Bigger for-profit companies need a company licence (see `LICENSE.md` in the repo).
- Ask Claude to keep one file per reel and reuse shared pieces (captions, character, end card) from a `components/` folder. That's how this page makes a new reel without starting from zero.
- Remotion's own docs: remotion.dev/docs. Repo: github.com/remotion-dev/remotion.

Sources this guide cites

http://remotion.dev/docs

http://github.com/remotion-dev/remotion