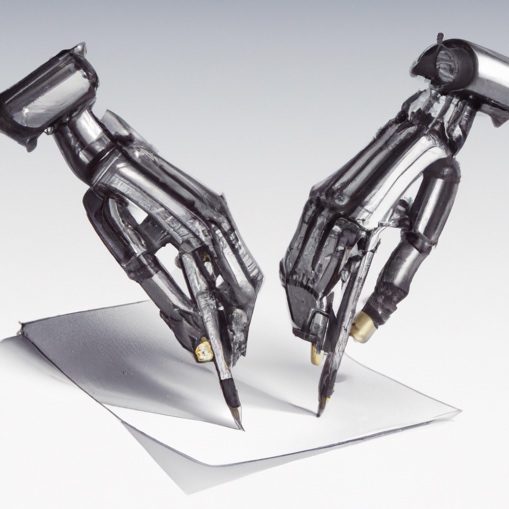
# Dall•e and chat-GPT helped me create a Facebook Instant Game



You can’t swing a dead cat without hitting some fanboy of chat-GPT. Every day, during my morning Internet fix I see some new article proclaiming GPT’s disruptive, work changing promise (or threat). Understood. My first exploration with the AI literally gave me the tingles. This was Google search on steroids!! It didn’t give links, it provided *answers*!

That excitement wore off fast. Between the repetitive structure of GPT’s responses and almost but not quite correct solutions to programming tasks it became clear I was interacting with an algorithm.

I’m more of a Dall•e fanboy if only for one reason, Dall•e is *fun*! I love submitting goofy prompts then waiting with bated breath for the thing to respond with some hilarious interpretation of my ask.

Hint: Past the 15 free credits, image generation costs 13¢ a pop. The service API starts at 1.6¢ per generated image. You can replace the Dall•e site with a Jupyter notebook containing around 10 lines of Python (ask Chat-GPT to write the code for you then correct its solution!). You get $18 of API credit when you first sign up. That’s over 1000 free image generations.

This got me thinking. Maybe, just maybe, other people would get as much pleasure from playing with Dall•e as I do. Hence was born the idea for the game “Call for Artists!”.

The experience of the game is straightforward. The user types a prompt into a text field, hits return (or press a button) and after bit the image appears. Toss in a few social features, for example sharing images or art competitions, and voila! A star is born!

As a free game it feels like it could catch on. As a source of revenue, it does not work. Image generation cost 1.6¢ a pop. A cursory search on Google for figures on Facebook ad revenue suggests I’d lose money. To make money I would have to require users to buy generation credits. To draw people in I’d give users a few teasers as a loss leader but ultimately… ugh.

So much for my easy source of unfettered riches. What started off as a fantasy source of income turned into an exploration of our recently disrupted software development ecosystem.

So, how does one go about creating a game? I know little to nothing about *playing* video games much less developing them. Video games destroyed pinball, so I made a lifetime pledge to avoid the murderous little snots. (Wolfenstein 3d broke my resolve temporarily) Now that you know I’m a tech grandpa you’ll understand my choice of gaming platform, Facebook Instant Games.

‘Call for Artists!’ would have about the same the level of sophistication with games already on the platform. Take a look sometime. It’s not like it would be rubbing shoulders with a ‘Call of Duty’.

Facebook gives a list of engines compatible with Instant Games. Sad to say Unity is not in this list. I did some exploration with Unity a while back and liked it a lot. It sits at a nice spot in the code to no-code spectrum of tools. Much can be done without code but when you need it, everything is accessible from C#.

Instead, I tried and ultimately used a tool called Construct 3. Don’t interpret my choice as a recommendation. C3 was fit for purpose, that’s all. It’s primarily no-code and has some support for Instant Games.

As far as I was concerned my biggest hurdle would be the game’s visual assets. I’m not a graphic designer nor will I ever pretend to be one. Since this project is to be a proof-of-concept placeholders will do. But neither can I use any old junk. Were I to change my mind and shop the idea around, folks would immediately write it off when they saw the screen.

After a day of contemplating whether to hire some low-cost gig worker to knock off a few resources the obvious hit me. Why not use Dall•e to create the images? Doh! The self-reference is beautiful.

As a test I asked Dall•e to create “A cartoon of Leonardo da Vinci holding a paintbrush”.

Logo

Description automatically generated

Nice!!! That is much better than I expected. Already I’m fired up and raring to forge onward. Next, I want a canvas, “A cartoon of a blank painting easel facing forward”

Shape, square

Description automatically generated

Not bad. It only needs the face chopped out and the overall image tweaked a bit.

We interrupt this irregularly scheduled program with a pitch for the *one and only* freeware software development project I’ve ever donated money to, Paint.net. Photoshop, GIMP? too complicated. Microsoft paint? Too primitive. Paint.net hits the sweet spot in the spectrum. If this non-artist can figure out how to use the tool and get decent results, anyone can.

Twenty minutes later, after a bit of cropping, erasing, coloring, and warping I got what I needed.

Shape, square

Description automatically generated

Continuing the DaVinci theme I decided the prompt input should looked like the player were writing on a medieval scroll.

“A cartoon drawing of a blank medieval scroll facing forward”

A picture containing indoor

Description automatically generated

It’ll do.

Lastly a few assorted and sundry items…

“A cartoon drawing of a horizontal brown rope” – a separator



“A cartoon of a long wooden plank” – the background of the submit button



“An artist's palette in cartoon style” – some eye candy

Icon

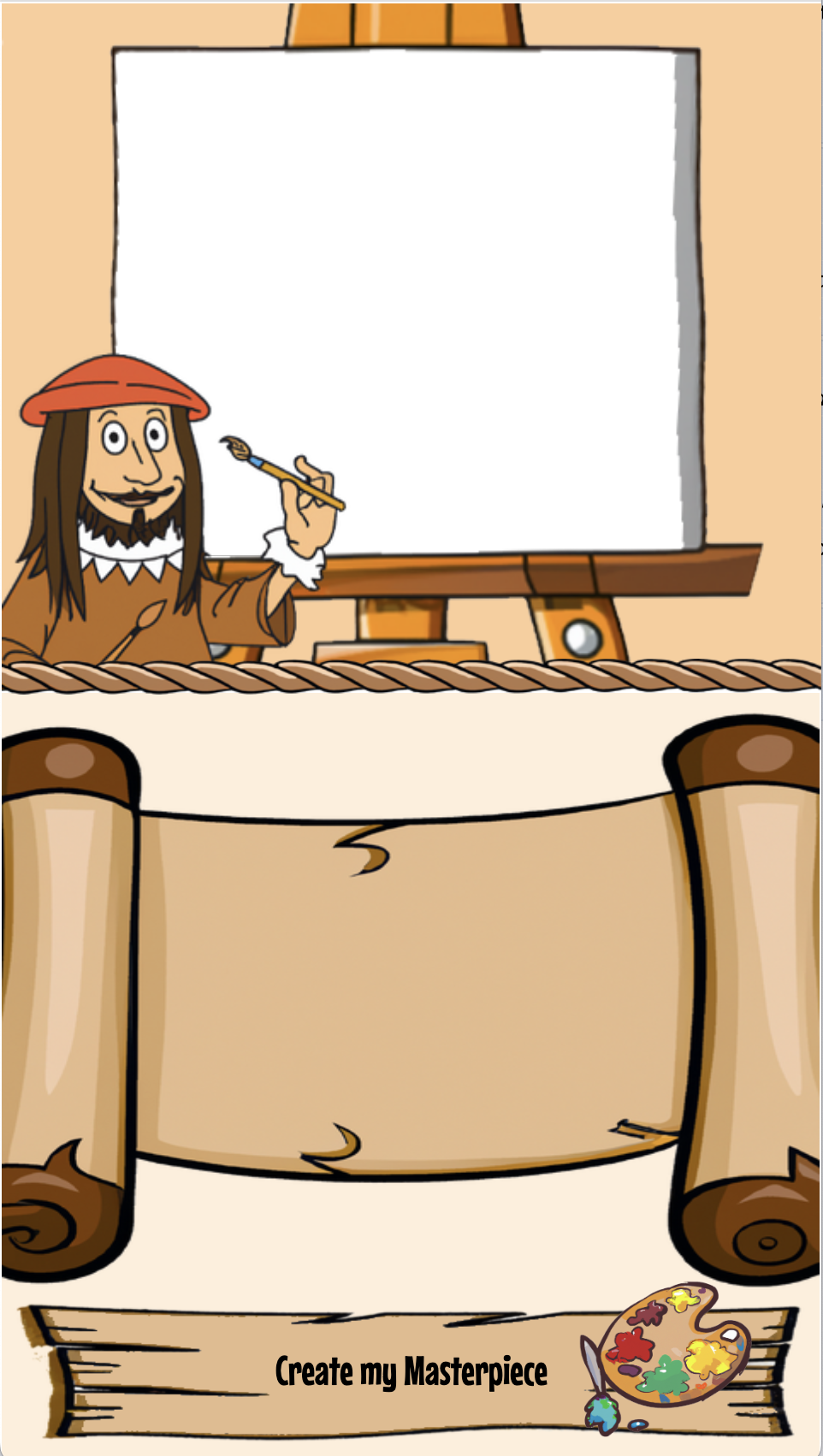
Description automatically generated with medium confidence

“Cartoon version of Leonardo da Vinci’s The Vitruvian Man” – for the spinner

A picture containing text

Description automatically generated

Armed with these assets I put the game screen together…



Not bad for a few hours’ effort. I would not be too embarrassed showing this to folks.

The logic driving the game is painfully simple. On pressing “Create my Masterpiece” the app sends the prompt over to a Back4App server which serves as a proxy to Dall•e. The response contains a link to the image which is then used as the src for an img element fitting exactly within the blank portion of the canvas.

About that proxy server. I did not want to my Open API app token accessible to any fool who knows how to bring up a browser’s developer tools. Back4App is easy to code against. It is even easier if you don’t write the code yourself!

Hey, chat-GPT, “Write a POST handler for the Parse cloud framework which receives a text prompt, calls the Dall·e service and returns the URL of the first generated image” …

The code it gave me almost worked. A few simple corrections to the JSON it sent in the POST request fixed it. This has been my experience generally with GPT’s code. The results *almost* work. This is not a bad thing IMHO. A tool which can do 95% of my work for me is better than one that puts me out of a job.

I built the logic backing the screen using Construct 3’s no code event programming GUI. The tool is not hard to use but neither is it inspiring. It was fit for purpose, period.

That’s all it took to get a working proof-of-concept. I recorded the game in action…

Let’s do the numbers. It took me five days to put the game together. Subtracting the 4½ days I spent scoping out tools and reading documentation leaves ½ day of real work. Experienced game developers would knock this off in far less time.

Without Dall•e I’d have wasted hours searching for visual resources. The proxy code I could have done without chatGPT but *with* it the level of effort was essentially zero. That is *so* cool.

In summary I’m convinced. Yes, programming is being disrupted, but in a good way. The way a lazy, impatient programmer like me appreciates. Going forward I shall welcome our new AI overlords with open arms, should they ever appear. Till then, I’ll gladly serve as a member of their fifth column.