The .dockerignore file

The .dockerignore file is the tool, that can help you to define the Docker build context you really need. Using this file, you can specify ignore rules and exceptions from these rules for files and folder, that won’t be included in the build context and thus won’t be packed into an archive and uploaded to the Docker server.

Why should you care?

Indeed, why should you care? Computers today are fast, networks are also pretty fast (hopefully) and storage is cheap. So, this “tax” may be not that big, right?

I will try to convince you, that you should care.

Reason #1: Docker image size

The world of software development is shifting lately towards continuous delivery, elastic infrastructure and microservice architecture.

Your systems are composed of multiple components (or microservices), each one of them running inside Linux container. There might be tens or hundreds of services and even more service instances. These service instances can be built and deployed independently of each other and this can be done for every single code commit. More than that, elastic infrastructure means that new compute nodes can be added or removed from the system and its microservices can move from node to node, to support scale or availability requirements. That means, your Docker images will be frequently built and transferred.

When you practice continuous delivery and microservice architecture, image size and image build time do matter. If it much faster to deploy a 5MB Docker image to 100 servers, than a 700MB image. It also helps local development as well.

Reason #2: Unintended secrets exposure

Not controlling your build context, can also lead to an unintended exposure of your code, commit history, and secrets (keys and credentials).

If you copy files into you Docker image with ADD . or COPY . command, you may unintentionally include your source files, whole git history (a .git folder), secret files (like .aws, .env, private keys), cache and other files not only into the Docker build context, but also into the final Docker image.

There are multiple Docker images currently available on DockerHub, that expose application source code, passwords, keys and credentials (for example Twitter Vine). Copying the .git folder in a Docker image by mistake is especially damaging. Tip: Always mention your .git folder in your .dockerignore file

Reason #3: The Docker build – cache invalidation

A common pattern is to inject an application’s entire codebase into an image using an instruction like this: