Frontend and backend

In software engineering, the terms **frontend** and **backend** (or sometimes referred to as **back end** or **back-end**) refer to the separation of concerns between the presentation layer (*frontend*), and the data access layer (*backend*) of a piece of software, or the physical infrastructure or hardware. In the client–server model, the client is usually considered the frontend and the server is usually considered the backend, even when some presentation work is actually done on the server itself.

Introduction

In software architecture, there may be many layers between the hardware and end user. The *front* is an abstraction, simplifying the underlying component by providing a user-friendly interface, while the *back* usually handles data storage and business logic.

In telecommunication, the *front* can be considered a device or service, while the *back* is the infrastructure that supports provision of service.

A rule of thumb is that the client-side (or "frontend") is any component manipulated by the user. The server-side (or "backend") code usually resides on the server, often far removed physically from the user.

Software definitions

- In content management systems, the terms *frontend* and *backend* may refer to the end-user facing views of the CMS and the administrative views, respectively.[1]
- In speech synthesis, the frontend refers to the part of the synthesis system that converts the input text into a symbolic phonetic representation, and the backend converts the symbolic phonetic representation into actual sounds.[2]
- For major computer subsystems, a graphical file manager is a frontend to the computer's file system, and a shell interfaces with the operating system. The frontend faces the user, and the backend launches the programs of the operating system in response.
- In compilers, the frontend translates a computer programming source code into an intermediate representation, and the backend works with the intermediate representation to produce code in a computer output language. The backend usually optimizes to produce code that runs faster. The front-end/back-end distinction can separate the parser section that deals with source code and the backend that generates code and optimizes. Some designs, such as GCC, offer choices between multiple frontends (parsing different source languages) or backends (generating code for different target processors).[3]
- Using the command-line interface (CLI) requires the acquisition of special terminology and memorization of commands, so a graphical user interface (GUI) acts as a frontend desktop environment instead.

Web development as an example

Another way to understand the difference between the two is to understand the knowledge required of a front-end vs. a back-end software developer. The list below focuses on web development as an example.

Both

- Version control tools such as Git, Mercurial, or Subversion
- File transfer tools and protocols such as FTP or rsync

Front-end focused

- Markup and web languages such as HTML, CSS, JavaScript, and ancillary libraries commonly used in those languages such as Sass or jQuery

- Asynchronous request handling and AJAX
- Single-page applications (with frameworks like React, Angular or Vue.js)
- Web performance (largest contentful paint, time to interactive, 60 FPS animations and interactions, memory usage, etc.)
- Responsive web design
- Cross-browser compatibility issues and workarounds
- End-to-end testing with a headless browser
- Build automation to transform and bundle JavaScript files, reduce images size... with tools like Webpack or Gulp.js
- Search engine optimization
- Accessibility concerns
- Basic usage of image editing tools such as GIMP or Photoshop
- User Interface

Back-end focused

- Scripting languages like PHP, Python, Ruby, Perl, Node.js, or Compiled languages like C#, Java or Go
- Automated testing frameworks for the language being used
- Application Data Access
- Application Business Logic
- Database administration
- Scalability
- High availability
- Security concerns, authentication and authorization
- Software Architecture
- Data transformation
- Backup methods and software

Note that both positions, despite possibly working on one product, have a very distinct set of skills.

API

The frontend communicates with backend through an API. In the case of web and mobile frontends, the API is often based on HTTP request/response. The API is sometimes designed using the "Backend for Frontend" (BFF) pattern, that serves responses to ease the processing on frontend side.[4]

Hardware definitions

In network computing, *frontend* can refer to any hardware that optimizes or protects network traffic.[5] It is called application front-end hardware because it is placed on the network's outward-facing frontend or boundary. Network traffic passes through the front-end hardware before entering the network.

In processor design, *frontend design* would be the initial description of the behavior of a circuit in a hardware description language such as Verilog, while *backend design* would be the process of mapping that behavior to physical transistors on a die.[6]

See also

- Front-end web development
- Client–server model
- Out-of-box experience
- Modular programming
- Observer pattern
- Publish–subscribe pattern
- Pull technology
- Push technology
- Remote procedure call

- Application program interface (API)

References

1. *Thapliyal, Vimal.* *"Difference Between Frontend and Backend MVC – Joomlatuts" (https://web.archive.org/web/20161230230237/http://joomlatuts.net/joomla-2-5/87-how-backend-model-view-controller-mvc-works-in-joomla/98-difference-between-frontend-and-backend-mvc)* *. joomlatuts.net. Archived from the original (http://joomlatuts.net/joomla-2-5/87-how-backend-model-view-controller-mvc-works-in-joomla/98-difference-between-frontend-and-backend-mvc)* *on 2016-12-30. Retrieved 2016-12-30.*
2. *Gutierrez--Osuna, Ricardo.* *"L18: Speech synthesis (backend)" (http://research.cs.tamu.edu/prism/lectures/sp/l18.pdf)* *(PDF). tamu.edu. Texas A&M University. Retrieved 2016-12-29.*
3. *Bin Muhammad, Rashid.* *"Operating Systems Notes" (http://www.personal.kent.edu/~rmuhamma/Compilers/MyCompiler/phase.htm)* *. www.personal.kent.edu. Kent State University. Retrieved 2016-12-30.*
4. *Wickramarachchi, Viduni (24 February 2021).* *"The BFF Pattern (Backend for Frontend): An Introduction" (https://blog.bitsrc.io/bff-pattern-backend-for-frontend-an-introduction-e4fa965128bf)* *. Bits and pieces. Retrieved 13 November 2021.*
5. *O'Dell, Mike.* *"Network Front-End Processors, Yet Again | June 2009 | Communications of the ACM" (http://cacm.acm.org/magazines/2009/6/28494-network-front-end-processors-yet-again/fulltext)* *. cacm.acm.org. Retrieved 2016-12-30.*
6. *"Front-End Design | Online Documentation for Altium Products" (http://techdocs.altium.com/display/ADOH/Front-End+Design)* *. techdocs.altium.com. Retrieved 2016-12-30.*

Last edited 6 months ago by P.T.Đ

WikipediA