

# Framework pro frontend - vlastnosti, výhody použití

---

## Frontend framework

- Je sada nástrojů a knihoven pro vývoj uživatelského rozhraní

### - Poskytuje:

- **Komponentový systém** = znovupoužitelné UI bloky
  - **Reaktivitu** = automatická aktualizace UI při změně dat
  - **Routing** = navigate mezi stránkami (SPA)
  - **State management** = správa stavu aplikace
  - **Build nástroje** = bundling, minifikace
- Umožňuje rychlejší vývoj a konzistentní UI

### - Přehled hlavních frameworků:

- **UI knihovny/frameworky:**
    - React
    - Vue.js
    - Angular
    - Svelte
  - **Meta-frameworky (přidávají SSR, routing a build systém nad základní UI knihovnou):**
    - Next.js
    - Nuxt
    - SvelteKit
    - Astro
- ### - Renderovací strategie:
- **CSR** (Client-Side Rendering):
    - JavaScript renderuje stránku v prohlížeči
    - Rychlá navigace, pomalý první load

- **SSR** (Server-Side Rendering):
  - Server vygeneruje HTML pro každý request
  - Rychlí první load, lepší SEO
- **SSG** (Static Site Generation):
  - HTML vygenerováno při buildu
  - Nejrychlejší, ale statický obsah
- **ISR** (Incremental Static Regeneration):
  - Kombinace SSG + revalidace
  - Next.js specifické
- **Kritéria pro výběr frameworku:**
  - **Technická:**
    - Výkon
    - Renderovací strategie
    - TypeScript podpora
    - Ekosystém knihoven
  - **Praktická:**
    - Dokumentace
    - Komunita
    - Trh práce

-

- **Přehled frameworků:**

• **React:**

- **Charakteristika:**

- **Virtual DOM** - diffing algoritmus —>
- **JSX** - HTML v JavaScriptu
- **Hooks** - funkcionální přístup
- Jednosměrný data flow
- Knihovna, ne framework

**Diffing algoritmus:**

- Porovnává starý a nový Virtual DOM strom a aplikuje pouze minimální změny do skutečného DOM
- Optimalizuje překreslování

- **Ekosystém:**

- **Next.js** - meta-framework
- **Redux /Zustand** - state management
- **React Query** - data fetching

**Výhody:**

- Největší ekosystém a komunita
- Obrovský pracovní trh
- React Native pro mobile
- 44% podíl mezi vývojáři

**Nevýhody:**

- „Just a library“ - must vybrat dal nástroje
- JSX křivka učení
- Častější breaking changes
- Boilerplate kód

- **Vue.js:**

- Charakteristika:

- **Reaktivní systém** - Proxy-based —>
    - **Single File Components** (.vue)
    - **Options API** nebo **Composition API**
    - **Template syntax** - HTML-like

- Proxy-based:

- **Vue 3** používá JavaScript Proxy ke sledování změn v datech
    - Když se data změní, UI se automaticky aktualizuje bez manuálního volání

- Ekosystém:

- **Nuxt** - meta framework
    - **Pinia** - oficiální state management
    - **Vite** - build tool (od Vue týmu)

- Výhody:

- Snadná křivka učení
    - Nejlepší dokumentace
    - Oficiální ekosystém (Router, Pinia)
    - Flexibilita (Options/Composition API)

- Nevýhody:

- Menší job market než React
    - Menší ekosystém knihoven
    - Méně enterprise adoption

- **Angular:**

- Charakteristika:

- **Full framework** - vše included
    - Typescript first
    - Dependency injection
    - **RxJS** - reaktivita

- Součásti:

- **Angular CLI** - scaffolding
    - **Angular Forms** - formuláře
    - **HttpClient** - HTTP request

- Výhody:

- Kompletní řešení - vše v jednom
    - Typescript
    - Skvělé pro velké týmy
    - Dlouhodobá podpora (LTS)

- Nevýhody:

- Strmá křivka učení
    - Verbose kód (boilerplate)
    - Velký bundle size (80KB+)
    - RxJS komplexita
    - Pomalejší vývoj pro MVP

- **Svelte**
  - Charakteristika:
    - **Kompilace** - žádný runtime
    - **Reaktivita** - \$: label
    - Nejmenší bundle size (2KB)
    - Vanilla JS output
    - Žádný Virtual DOM
  - Ekosystém:
    - **SvelteKit** - meta framework
    - **Svelte stores** - state management
    - **Svelte actions** - direktivy

#### Výhody:

- Nejmenší bundle size
- Nejlepší runtime výkon
- Jednoduchá syntaxe
- Žádný boilerplate
- Nejvyšší spokojenost

#### Nevýhody:

- Menší ekosystém
- Méně job opportunities
- Méně zralý než React/Vue
- Kompilátor = vendor lock-in
- Menší komunita

- **Meta frameworky (UI knihovna + full stack):**

- Server-Side rendering (SSR)
- Static Site Generation (SSG)
- File based routing
- API routes - backend v jednom projektu
- Optimalizace - code splitting, lazy loading
- Deployment - edge functions, serverless

- **Next.js:**

• **Features:**

- App Router
- SSR, SSG, ISR
- API routes - backend endpoints
- Image optimization
- Middleware

- **Nuxt:**

- **Nuxt 3 přinesl Nitro - univerzální server engine pro ether i Node.js**

• **Features:**

- Auto-imports
- File-based routing
- SSR, SSG, hybrid
- Nitro server - univerzální backend
- Nuxt Modules - rozšíření

• **DX features:**

- Nuxt DevTools
- Zero-config Typescript
- Built-in SEO utils
- Nuxt Content (CMS)

- **SvelteKit:**

- Oficiální Svelte meta-framework
- File-based routing
- SSR, SSG, prerendering
- Adaptery pro různé platformy
- Nejmenší bundle size

- **Astro:**

- Islands architecture
- Zero JS by default
- Podporuje React, Vue, Svelte
- Ideální pro content sites
- Partial hydration