

# Excalidraw

Teknisk Arkitektur & Implementation

En djupdykning i modern webbutveckling

# Vad är Excalidraw?

- ✓ Webbaserad ritapplikation (whiteboard)
- ✓ Skapar handritade diagram och skisser
- ✓ Gratis och open-source
- ✓ Ingen installation behövs
- ✓ Möjlighet att samarbeta i realtid
- ✓ +1000 automatiserade tester

# Tech Stack - Vad används?

Teknologi	Version	Syfte
<b>React</b>	19.0.10	UI Framework
<b>TypeScript</b>	5.9.3	Type Safety
<b>Vite</b>	5.0.12	Build Tool
<b>Canvas 2D</b>	-	Rendering
<b>Vitest</b>	3.0.6	Testing

# Arkitektur - Olika Lager

- **React Frontend:** Toolbar, Canvas, Panels, Dialogs
- **State Management:** Elements, AppState, History, Scenes
- **Rendering Engine:** Canvas 2D API, WebGL acceleration
- **Core Logic:** Element manipulation, Collision detection
- **Storage:** localStorage, IndexedDB, CloudStorage

**Varför lager?** Separation of concerns - varje lager gör en sak bra

# Varför React?



## Komponentbaserad

Återanvändbar kod, enkel att organisera



## Virtual DOM

Effektiv rendering av UI-ändringar



## Community

Stor community, många bibliotek

# Rendering: Canvas vs SVG

## Canvas (Vald)

- ✓ Högt performance
- ✓ Perfekt för ritprogram
- ✓ GPU acceleration
- ✓ Handritad känsla

## SVG (Ej vald)

- ✗ Långsammare med många element
- ✗ Svårt att implementera effekter
- ✓ Bättre för vektor-grafik
- ✓ Enklare manipulation

# Canvas Rendering - Pipeline

```
// 1. Hämta context const ctx = canvas.getContext('2d'); // 2. Rensa ctx.clearRect(0, 0, width, height); // 3. Applicera transformationer ctx.transform(...); // 4. Rita alla element  
elements.forEach(el => { ctx.fillRect(el.x, el.y, el.width, el.height); });
```

**Två canvas-lager:** Interactive (musinteraktion) + Static (innehåll)






# Element - Datastruktur

```
interface Element { id: string; type: 'rectangle' | 'diamond' | 'ellipse' | 'arrow' | 'text'; x: number; y: number; width: number; height: number; backgroundColor: string; strokeColor: string; angle: number; // rotation  
locked: boolean; text: string; // för text-element }
```

Varje form är ett objekt med position, stil, och egenskaper



# Interaction Event Loop

-  **User Action:** Klick, drag, tangent
-  **Event Listener:** onPointerDown, onPointerMove, onPointerUp
-  **Calculate:** Vad ändrade sig?
-  **Update State:** Element, AppState
-  **Re-render:** Rita på canvas

# Exempel: Drag an Element

```
// Pointer Down onPointerDown = (event) => { state.draggedElement = getElementAtPosition(event.x, event.y); }  
// Pointer Move onPointerMove = (event) => { if (state.draggedElement) { state.draggedElement.x = event.x -  
offset.x; state.draggedElement.y = event.y - offset.y; render(); } } // Pointer Up onPointerUp = () => {  
addToHistory(state); // För undo }
```

# Undo/Redo System

```
class History { entries = []; currentIndex = -1; addEntry(elements, appState) { this.entries =  
this.entries.slice(0, this.currentIndex + 1); this.entries.push({ elements, appState });  
this.currentIndex++; } undo() { this.currentIndex--; } redo() { this.currentIndex++; } }
```

Spara snapshot av varje tillstånd → enkelt undo/redo

# Storage & Persistence

## localStorage

5-10MB. Snabb. Synkron. Lokalt

## IndexedDB

50MB+. Asynkron. Ingen blockering

## Cloud

Obegränsad. Samarbete. Synk mellan enheter

**Excalidraw använder:** localStorage för små diagrams, IndexedDB för större




# Testing - 1079 Tests!

- **Vitest:** Test runner (snabb, moderna)
- **@testing-library:** Test som användare gör
- **vitest-canvas-mock:** Mock Canvas API för tests

## **Test-resultat:**

✓ 964 tests passar  
68 snapshot mismatches (ej kritiskt)  
46 skipped

# Test Pyramid

-  **Unit Tests:** Enskilda funktioner (många, snabba)
-  **Integration Tests:** Flera komponenter tillsammans (medel)
-  **E2E Tests:** Fullständiga användar-scenarier (få, långsamma)

**Excalidraw fokuserar på:** Integration + Unit (viktigast för UI)

# Deployment - 3 Alternativ

## 1 Web App

Deploy till GitHub Pages, Netlify

## 2 NPM Package

Andra appar kan importera det

## 3 Docker

Container, server-deployment

# Monorepo - Projektstruktur

```
excalidraw/ ├── excalidraw-app/ # Web app ├── packages/ | ├── common/ # Shared utilities | ├── element/ #  
Element types | ├── excalidraw/ # Core library | ├── math/ # Math utilities | └── utils/ # Helper functions └──  
examples/ # Example implementations
```

**Fördel:** Shared code, koordinerad versionering



# Performance Optimizations

- ✓ **Memoization:** Cache funktions-resultat
- ✓ **Selective Rendering:** Rita bara synliga element
- ✓ **Event Debouncing:** Begränsa event-frekvens
- ✓ **Virtual Scrolling:** Rendera bara synliga i listor
- ✓ **Two Canvas Layers:** Separate interactive och static

# Architecture Decisions & Tradeoffs

Val	Anledning	Tradeoff
Canvas	Performance	Svårare att implementera vissa features
useReducer	Modern React	Inte optimerad för mycket stora states
Monorepo	Shared code	Mer komplext setup

# Development Workflow

```
# Installera yarn install # Starta dev-server (hot reload) yarn start # Köra tester yarn  
test:app # Code quality yarn test:code # ESLint yarn test:typecheck # TypeScript yarn test:all  
# Allt # Build för production yarn build
```

# Scaling - 10,000+ Element

## Problem







- Långsam rendering
- Memory leak
- Laggy interaktion

## Lösningar

- Spatial indexing
- Culling (rita bara synliga)
- Throttling
- Worker threads

20

## Key Learnings

-  Test-setup är kritiskt
-  Coverage  $\neq$  Quality
-  Exploratory testing bekräftar teorierna
-  Architecture decisions påverkar allt
-  Performance kräver konstant focus
-  Open-source = högt kodkvalitet

# Sammanfattning

Excalidraw är byggt på moderna web-teknologier  
med focus på performance, testning, och user experience

En utmärkt exempel på professionell webbutveckling