Next.js Cheatsheet

1 Pages and Routing

Next.js uses file-based routing. Files in the `pages/` directory become routes.

```
// pages/index.js
export default function Home() {
  return <h1>Welcome to Next.js!</h1>;
}

// pages/about.js
export default function About() {
  return <h1>About Page</h1>;
}
// Accessible at / and /about
```

2 Dynamic Routes

Create dynamic routes using file names with square brackets.

```
// pages/users/[id].js
export default function User({ id }) {
  return <h1>User ID: {id}</h1>;
}
```

3 Static Site Generation (SSG)

Generate pages at build time with `getStaticProps`.

```
// pages/index.js
export async function getStaticProps() {
  const data = await fetch('https://api.example.com/data').then(res => res.json
      ());
  return { props: { data } };
}

export default function Home({ data }) {
  return <div>{data.title}</div>;
}
```

4 Server-Side Rendering (SSR)

Render pages on each request with `getServerSideProps`.

```
// pages/post.js
export async function getServerSideProps(context) {
  const { id } = context.params;
  const data = await fetch(`https://api.example.com/post/${id}`).then(res =>
      res.json());
  return { props: { data } };
}

```

```
export default function Post({ data }) {
  return <h1>{data.title}</h1>;
}
```

5 API Routes

Create serverless API endpoints in `pages/api/`.

```
// pages/api/hello.js
export default function handler(req, res) {
  if (req.method === 'GET') {
    res.status(200).json({ message: 'Hello, API!' });
  }
}
```

6 useRouter

Access routing information and navigate programmatically.

```
import { useRouter } from 'next/router';

export default function Page() {
  const router = useRouter();
  const { id } = router.query; // Access dynamic params
  return (
    <button onClick={() => router.push('/home')}>
      Go to Home
    </button>
  );
}
```

7 Built-in CSS Support

Style components using CSS modules or global styles.

```
// styles/Home.module.css
.container {
  padding: 20px;
}

// pages/index.js
import styles from '../styles/Home.module.css';

export default function Home() {
  return <div className={styles.container}>Styled Content</div>;
}

// Global CSS: styles/global.css
body { margin: 0; }
```

8 Image Optimization

Use Next.js `Image` component for optimized images.

```
import Image from 'next/image';

export default function Home() {
  return (
    <Image
```

```
      src="/example.jpg"
      alt="Example"
      width={500}
      height={300}
      priority
    />
  );
}
```

9 Deployment Basics

Deploy Next.js apps to platforms like Vercel.

```
# Install Vercel CLI
npm install -g vercel

# Deploy
vercel

# Build for production
npm run build
npm run start
```

10 Best Practices

- Use SSG (`getStaticProps`) for static content to improve performance.
- Use SSR (`getServerSideProps`) for dynamic, request-time data.
- Leverage `useRouter` for client-side navigation.
- Use CSS modules for scoped styling.
- Optimize images with `next/image` to reduce load times.
- Deploy to Vercel for seamless scaling and automatic optimizations.