Nuxt.js

Starting with nuxt.js

```
$ npm install -g vue-cli
```

```
$ vue init nuxt-community/starter-template [프로젝트 이름]
```

```
$ cd [생성된 프로젝트 디렉토리]
$ npm install # or yarn  # npm 또는 yarn을 이용하여 의존성 모듈 설치
$ npm run dev  # nuxt.js 실행
```

프로젝트 구조

- **nuxt.config.js**; nuxt.js의 설정 정보 파일. build, cache, css, dev, env, generate, head, loading, plugin, rootDir, router, srcDir, transition의 설정
- **package.json**; 프로젝트의 의존성 모듈과 각종 정보, 스크립트를 포함하는 파일
- **assets**; SASS, LESS나 빌드가 필요한 JavaScript 파일이 포함되는 디렉토리
- **components**; nuxt.js에서 사용되는 컴포넌트들을 포함하는 디렉토리, .vue 파일
- **layouts**; nuxt.js에서 사용하는 기본 레이아웃을 포함하는 디록토리
- **middleware**; 레이아웃, 페이지가 렌더링되기 전에 실행되는 파일이 정의되는 디렉토리.
- **pages**; 일반적인 vue 파일을 포함하는 디렉토리. asyncData, fetch, head, layout, transition, scrollToTop, validate, middleware 속성 사용 가능
- **plugins**; vue로 만든 웹 애플리케이션이 생성되기 전에 실행시키고 싶은 js 파일을 포함하는 디렉토리.
- **static**; 정적 파일, 이미지 파일이나 각종 CSS, javaScript 같은 정적 파일을 포함하는 디렉토리.
- **store**; 상태(데이터) 관리를 도와주는 라이브러레인 vuex가 포함된 디렉토리.

pages

```
<template>

</template>

<script>
export default {

  // vue 인스턴스가 생성되기 전 store 초기화
  fetch({store, params}) {
  },

 // 인스턴스가 생성될 때 호출
```

```
  created() {
  },
 // 비동기적으로 데이터 생성 json 형태로 반환하면 data()와 머지됨.
  asyncData(context) {
    return {}
  },
 // 데이터 생성
  data() {
    return {}
  },
  // (optional) 해당 페이지 <head> 커스터마이징
  head() {
    return {
      title: '',
      meta: [
        { hid: '고유값', name: '설명', content: '커스텀 설명'}
      ]
    }
  },
 // 해당 파일에 동작시킬 메소드 정의
  methods: {
  },
 // 복잡한 계산식일 때 computed를 이용
  computed: {
  },
 // 외부 컴포넌트 추가 {a, b, c}의 형태로 작성. 각각의 컴포넌트는 import a from 'a'로 가져온다
  components: {
  },
  // (optional), layout 설정, 기본 레이아웃 사용시 필요없음
  layout: '',
  // (optional), 미들웨어 설정, middleware: 'authenticated'는 middleware 디렉토리의 authenticated.js 파일 실행
  middleware: '',
  // (optional), 최상단 true, 최하단 false
  scrollToTop: true || false
}
</script>
```

- asyncData 메소드;
- fetch 메소드;
- head;
- layout;
- middleware;
- scrollToTop;

- sytle; 해당 페이지에서만 적용할 스타일 <style scoped></style>

설정 파일

- nuxt.config.js
  - cache
  - loading

Routes

- 정적라우트
- 동적라우트; _id.vue 등 this.$route.params.[_이름]
- 페이지 이동; <nuxt-link> 컴포넌트 이용. 여러 줄일경우 등으로 묶어주어야 한다. 하나의 루투 요소로 포함.

layouts

- default 레이아웃; default.vue
- error 레이아웃; error.vue

Components

- 컴포넌트 생성; import ~ from
- 컴포넌트로 데이터 넘기기; props를 이용
- 컴포넌트로 함수 넘기기;
- props 검사; props로 전달되는 인자 타입 7가지: String, Number, Boolean, Function, Object, Array, Symbol

```
<script>
export default {
  props: {
    p1: {Number, String},
    p2: {
      tyep: String,
      required: true
    },
    p3: {
      type: Number,
      default: 100
    },
    p4: {
      type: Object,
      default: function() {
        return { message: 'hello world' }
      }
    },
    p5: {
      validator: function(value) {
```

```
        return value > 0
      }
    }
}
</script>
```

- props의 한계; 단방향 바인딩.

Vuex store를 이용한 데이터 관리

- Vuex store; this.$store 저장소, mutations 메서드로 상태 변경
- mapState 사용
- fetch 메소드
- vuex store 모듈 형태 제작; 규모가 커짐에 따라 상태를 독립적으로 관리.

express와 nuxt.js 같이 사용

- nuxt.js 프로젝트 생성

```
$ vue init nuxt-community/starter-template nuxt_with_express

$ npm install
$ npm install -s express
```

- express 앱 추가
- nuxt with express 보일러 플레이트

```
$ vue init nuxt-community/starter-template [프로젝트 명]  # nuxt로만 이루어진 웹 애플리케이션 프로젝트 생성
$ vue init nuxt-community/express-template [프로젝트 명]  # express 앱 자동 추가
```

```
$ npm install
$ npm run build
$ npm run start
$ npm run dev
```

- 에러 발생 시 package.json 파일을 수정하거나 server/index.js 파일을 ES 버전에 맞게 수정

References

- Nuxt.js 란? 개념과 예제 (SSR, CSR, Universal, asyncData, fetch...)
- [Nuxt Admin] 4. Nuxt에서 컴포넌트 사용
- Nuxt.js 기초 강좌 : 페이지와 컴포넌트
- Creating Dynamic Routes in a Nuxt Application
- NUXT 기초 강좌 : 변수 선언 방법, 함수 사용법
- How to get current route name in Nuxt.js?
- NUXT로 혼자 웹사이트 만들기 4 플러그인 설치

- Choosing the right Node.js Framework: Next, Nuxt, Nest?
- Nuxt.js 란? (feat. SSR, CSR)

From:
https://theta5912.net/ - **reth**

Permanent link:
https://theta5912.net/doku.php?id=public:computer:nuxtjs

Last update: **2022/04/28 15:30**