

시작하기 및 CLI

Sails.js는 Ruby on Rails 스타일의 MVC 패턴과 실시간 WebSockets 기능을 통합한 Node.js 엔터프라이즈 프레임워크입니다.

- CLI 설치: `npm install -g sails`
- 새 프로젝트 생성: `sails new my-app`
- 서버 실행: `sails lift`
- API 생성 (모델 및 컨트롤러): `sails generate api user`
- 단일 액션 생성: `sails generate action user/login`
- 모델 생성: `sails generate model post`
- 대화형 콘솔: `sails console`

라우팅 및 블루프린트 (Routing & Blueprints)

커스텀 라우트 (config/routes.js)

```
module.exports.routes = {
  'GET /': { view: 'pages/homepage' },
  'POST /api/v1/users':
    'UserController.create',
  'GET /api/v1/users/:id': 'user/find-one', // Actions2 문법
  'PUT /api/v1/posts/:id': { action:
    'post/update', cors: true },
};
```

블루프린트 API (Blueprints)

모델 생성 시 별도 코드 없이 표준 REST API가 자동 제공됩니다.

- GET /user: 모든 레코드 조회 (파라미터 필터링 지원)
- GET /user/:id: 특정 ID 레코드 조회
- POST /user: 새 레코드 생성
- PATCH /user/:id: 레코드 부분 수정
- DELETE /user/:id: 레코드 삭제

워터라인 ORM (Waterline ORM & Models)

api/models/User.js

```
module.exports = {
  attributes: {
    email: { type: 'string', required:
true, unique: true, isEmail: true },
```

```
password: { type: 'string',
required: true, protect: true },
age: { type: 'number', min: 0 },
// 1:N 관계 정의
posts: { collection: 'post', via:
'author' },
},
};
```

모델 쿼리 (Queries)

- `const users = await User.find({ age: { '≥': 21 } }).sort('createdAt DESC').limit(10);`
- `const user = await User.findOne({ email });`
- `const newUser = await User.create({ email, password }).fetch();`
- `const updated = await User.updateOne({ id }).set({ age: 22 });`
- `await User.destroyOne({ id });`
- `const authorWithPosts = await User.findOne({ id }).populate('posts');`

컨트롤러 및 액션 (Actions2)

api/controllers/user/find-one.js

```
module.exports = {
  friendlyName: 'Find one user',
  description: '사용자 단일 정보를 반환합니다.',
```

```
inputs: {
  id: { type: 'number', required:
true, description: '사용자 ID' },
},
```

```
exits: {
  success: { responseType: 'view',
viewTemplatePath: 'pages/user/
profile' },
  notFound: { responseType:
'notFound' },
},
```

```
fn: async function (inputs, exits) {
  const user = await
User.findOne({ id: inputs.id });
  if (!user) { return
```

```
exits.notFound(); }
  return exits.success({ user });
},
};
```

정책 및 접근 제어 (Policies)

컨트롤러나 액션 실행 전 인가를 수행하는 미들웨어.

```
// api/policies/isLoggedIn.js
module.exports = async function (req,
res, proceed) {
  if (req.session.userId) {
    return proceed();
  }
  return res.forbidden();
};
```

```
// config/policies.js 매핑
module.exports.policies = {
  '*': 'isLoggedIn',
  'user/login': true, // 로그인 액션은 공
개
};
```

실시간 웹소켓 (Sails Sockets)

- 클라이언트 수신:

```
io.socket.get('/user', (data, jwr) =>
{ console.log(data); });
io.socket.on('user', (event) =>
{ console.log('변경 감지:', event); });
```

- 서버 브로드캐스트:

```
sails.sockets.broadcast('roomName',
'message', { text: 'Hello' });
```

헬퍼 함수 (Sails Helpers)

- 호출: `sails.helpers.sendEmail.with({ to, subject, text });` `await`