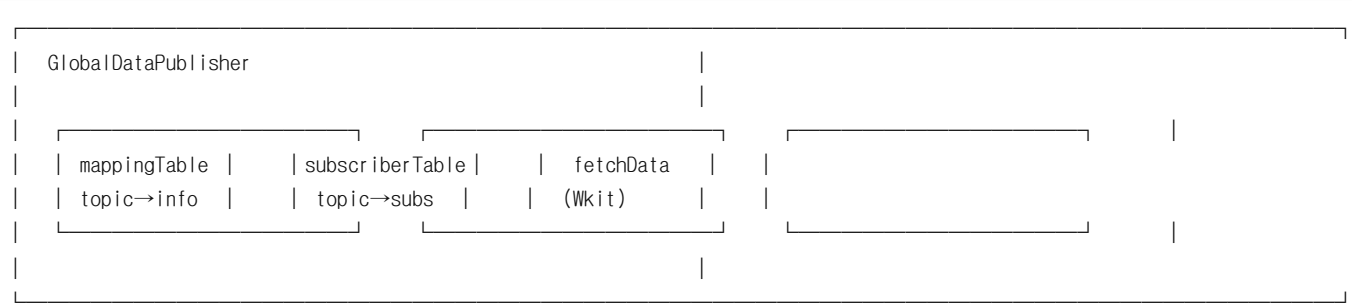
GlobalDataPublisher API Reference

전역 데이터 Pub/Sub 시스템. 여러 컴포넌트가 동일한 데이터 소스를 공유할 때 사용.

개요



흐름:
1. registerMapping(topic, datasetInfo) - 데이터 소스 등록
2. subscribe(topic, instance, handler) - 구독자 등록
3. fetchAndPublish(topic, page) - 데이터 fetch → 구독자에게 배포

API

registerMapping({ topic, datasetInfo })

토픽과 데이터 소스 매핑 등록.

```
/**
 * @param {Object} config
 * @param {string} config.topic - 토픽 이름
 * @param {Object} config.datasetInfo - 데이터 소스 정보
 * @param {string} config.datasetInfo.datasetName - 데이터셋 이름
 * @param {Object} config.datasetInfo.param - 기본 파라미터
 * @returns {Object} - 등록된 매핑 정보
 */
GlobalDataPublisher.registerMapping({
    topic: 'users',
    datasetInfo: {
        datasetName: 'userapi',
        param: { limit: 10 }
    }
});
```

---

unregisterMapping(topic)

토픽 매핑 해제.

```
GlobalDataPublisher.unregisterMapping('users');
```

---

subscribe(topic, instance, handler)

토픽 구독.

```
/**
 * @param {string} topic - 토픽 이름
 * @param {Object} instance - 구독하는 인스턴스 (this)
 * @param {Function} handler - 데이터 수신 핸들러
 */
GlobalDataPublisher.subscribe('users', this, function(data) {
    // data: API 응답 데이터
    this.renderUsers(data);
});
```

핸들러 컨텍스트: `handler.call(instance, data)` - this가 instance로 바인딩됨.

---

unsubscribe(topic, instance)

토픽 구독 해제.

```
GlobalDataPublisher.unsubscribe('users', this);
```

---

fetchAndPublish(topic, page, paramUpdates?)

데이터 fetch 후 모든 구독자에게 배포.

```
/**
 * @param {string} topic - 토픽 이름
 * @param {Object} page - 페이지 객체 (dataService 접근용)
 * @param {Object} [paramUpdates] - 기본 param에 병합할 추가 파라미터
 */

// 기본 파라미터로 fetch
await GlobalDataPublisher.fetchAndPublish('users', this.page);

// 파라미터 오버라이드
await GlobalDataPublisher.fetchAndPublish('users', this.page, {
    limit: 20,
    offset: 10
});
```

파라미터 병합:

```
// datasetInfo.param = { limit: 10 }
// paramUpdates = { offset: 10 }
// 결과: { limit: 10, offset: 10 }
```

---

getGlobalMappingSchema(options?)

기본 매핑 스키마 생성 헬퍼.

```
/**
 * @param {Object} [options]
 * @param {string} [options.topic='weather']
 * @param {Object} [options.datasetInfo]
 * @returns {Object} - { topic, datasetInfo }
 */
```

```
const schema = GlobalDataPublisher.getGlobalMappingSchema({
    topic: 'myTopic',
    datasetInfo: {
        datasetName: 'myapi',
        param: { id: 1 }
    }
});
```

사용 패턴

초기화 (페이지 또는 컴포넌트)

```
// 1. 매핑 등록
this.globalDataMappings = [
    {
        topic: 'users',
        datasetInfo: {
            datasetName: 'userapi',
            param: { limit: 10 }
        }
    },
    {
        topic: 'products',
        datasetInfo: {
            datasetName: 'productapi',
            param: { category: 'all' }
        }
    }
];

fx.each(
    (mapping) => GlobalDataPublisher.registerMapping(mapping),
    this.globalDataMappings
);
```

구독 (컴포넌트)

```
// 구독 등록
this.subscriptions = {
    users: [this.renderUserList, this.updateUserCount],
    products: [this.renderProductGrid]
};

fx.each(([topic, handlers]) => {
    fx.each((handler) => {
        GlobalDataPublisher.subscribe(topic, this, handler);
    }, handlers);
}, Object.entries(this.subscriptions));
```

데이터 요청

```
// 초기 로드
await GlobalDataPublisher.fetchAndPublish('users', this.page);

// 페이지네이션
await GlobalDataPublisher.fetchAndPublish('users', this.page, {
    offset: this.currentPage * 10
});

// 필터링
await GlobalDataPublisher.fetchAndPublish('products', this.page, {
    category: 'electronics'
});
```

정리 (destroy)

```
// 구독 해제
fx.each((topic) => {
    GlobalDataPublisher.unsubscribe(topic, this);
}, Object.keys(this.subscriptions));

// 필요 시 매핑 해제
fx.each((mapping) => {
    GlobalDataPublisher.unregisterMapping(mapping.topic);
}, this.globalDataMappings);
```

내부 구조

저장소	타입	용도
mappingTable	Map<topic, datasetInfo>	토픽 → 데이터소스 매핑
subscriberTable	Map<topic, Set<{instance, handler}>>	토픽 → 구독자 목록

에러 처리

```
try {
    await GlobalDataPublisher.fetchAndPublish('users', this.page);
} catch (error) {
    // fetch 실패 시 에러 throw
    console.error('데이터 로드 실패:', error);
}
```

콘솔 경고:

- `[GlobalDataPublisher] 등록되지 않은 topic: xxx` - 미등록 토픽으로 fetch 시도

관련 문서

- WKIT_API.md - fetchData 구현
- DEFAULT_JS_NAMING.md - datasetInfo 형태 규칙