Echarts (Mixed Line Bar)

ECharts Mixed Chart (Bar + Line) 구현 가이드

📌 시작하기 전에

- 자바스크립트를 몰라도 괜찮습니다
- 아래 코드를 **전체 복사** 후 필요한 부분만 수정하세요
  - (가장 하단의 `clearInterval(this.myInterval)`은 destroy 탭에)
- `this.`는 각 컴포넌트를 구분하는 표시이므로 그대로 두세요

🔧 전체 코드 구조

```
// 1. 기본 함수 정의
this.createChart = (element) => echarts.init(element);

this.settingChart = fx.curry((chart, option) => chart.setOption(option));

this.setXAxis = fx.curry((option, response) => {
    option.xAxis = {
        type: 'category',
        data: response.categories,
    };
    return response;
});

// 이중 Y축 설정 (왼쪽: 금액, 오른쪽: 퍼센트)
this.setDualYAxis = fx.curry((option, response) => {
    option.yAxis = [
        {
            type: 'value',
            name: 'Amount',
            position: 'left'
        },
```

```
        {
            type: 'value',
            name: 'Growth Rate (%)',
            position: 'right',
            axisLabel: {
                formatter: '{value}%'
            }
        }
    ];
    return response;
});

// Bar와 Line 혼합 시리즈
this.setMixedSeries = fx.curry((option, response) => {
    option.series = response.series.map(serie => ({
        name: serie.name,
        type: serie.type,
        data: serie.data,
        yAxisIndex: serie.yAxisIndex || 0,  // 기본값 0 (왼쪽 Y축)
        ...(serie.type === 'line' && {
            smooth: true,
            symbol: 'circle',
            symbolSize: 6
        }),
        ...(serie.type === 'bar' && {
            barWidth: '20%'
        })
    }));
    return response;
});

this.setTooltip = fx.curry((option, response) => {
    option.tooltip = {
        trigger: 'axis',
        axisPointer: {
            type: 'cross'
        }
    };
```

```
    option.legend = {
        data: response.series.map(s => s.name),
        top: 10
    };
    return response;
});

this.getChartData = async apiArguments => {
    const { response } = await WKit.fetchData(...Object.values(apiArgument
s)).catch(console.error);
    return response;
};

// 2. 차트 생성 및 설정
const chartElement = this.element.querySelector('#echarts');
this.myMixedChart = this.createChart(chartElement);

const chartOption = {};

// 🔴 API 연결 정보 (수정 필요)
const apiArguments = {
    page: this.page,
    datasetName: 'api_combined_chart',    // ← 여러분의 API 이름으로 변경
    params: { port: 3000, months: 12 }    // ← 필요한 파라미터로 변경
};

// 3. 차트 로드 함수
const loadChart = () => fx.go(
    this.getChartData(apiArguments),
    this.setXAxis(chartOption),
    this.setDualYAxis(chartOption),
    this.setMixedSeries(chartOption),
    this.setTooltip(chartOption),
    (_) => this.settingChart(this.myMixedChart, chartOption)
);

loadChart();
```

```
// 🟡 자동 새로고침 설정 (필요시 수정)
this.REFRESH_INTERVAL = 5000;  // 5초마다 새로고침
this.myInterval = setInterval(loadChart, this.REFRESH_INTERVAL);

// 4. 정리 (destroy 탭에 위치)
clearInterval(this.myInterval);
```

📊 데이터 구조 이해하기

API 응답 구조:

```
{
 "categories": [
  "Sep", "Oct", "Nov", "Dec", "Jan", "Feb",
  "Mar", "Apr", "May", "Jun", "Jul", "Aug"
 ],
 "series": [
  {
   "name": "Revenue",
   "type": "bar",            // Bar 차트
   "data": [2493, 3972, 4924, 2707, 2631, 2345, ...]
  },
  {
   "name": "Profit",
   "type": "bar",            // Bar 차트
   "data": [834, 955, 261, 705, 598, 952, ...]
  },
  {
   "name": "Growth Rate",
   "type": "line",           // Line 차트
   "yAxisIndex": 1,          // 오른쪽 Y축 사용
   "data": [8, 4, -1, 28, 23, 13, ...]
  }
 ]
}
```

중요 포인트:

- `categories`: X축 라벨 (월, 일, 카테고리 등)
- `series`: 차트 데이터 배열
  - `type`: 'bar' 또는 'line'
  - `yAxisIndex`: 0 = 왼쪽 Y축 (기본값), 1 = 오른쪽 Y축
- Bar 차트는 주로 **절대값** (매출, 이익)
- Line 차트는 주로 **비율/추세** (성장률, 변화율)

🎯 데이터 흐름 설명

`loadChart` 함수의 동작:

```
const loadChart = () => fx.go(
    this.getChartData(apiArguments),     // 1단계: API 호출
    this.setXAxis(chartOption),          // 2단계: X축 설정
    this.setDualYAxis(chartOption),      // 3단계: 이중 Y축 설정
    this.setMixedSeries(chartOption),    // 4단계: Bar + Line 데이터 설정
    this.setTooltip(chartOption),        // 5단계: 툴팁과 범례 설정
    (_) => this.settingChart(this.myMixedChart, chartOption)  // 6단계: 차트 그리기
)
```

단계별 설명:

1. **API 데이터 가져오기**: categories와 series 포함된 response
2. **X축 설정**: 월/날짜 등 카테고리 배치
3. **이중 Y축 설정**: 왼쪽(금액), 오른쪽(퍼센트)
4. **혼합 시리즈 설정**: Bar와 Line 각각 처리
5. **툴팁/범례 설정**: 마우스 호버 시 정보 표시
6. **차트 렌더링**: 완성된 옵션으로 차트 그리기

✏️ 수정 가능한 부분

1. API 정보 변경 (🔴 필수)

```
const apiArguments = {
    page: this.page,
    datasetName: 'sales_analysis',      // API 이름 변경
    params: {
        port: 3000,
        months: 6,                      // 6개월 데이터
        department: 'sales',            // 부서별 필터
        includeGrowth: true             // 성장률 포함
    }
}
```

2. Y축 이름/단위 변경 (🟡 선택)

```
this.setDualYAxis = fx.curry((option, response) => {
    option.yAxis = [
        {
            type: 'value',
            name: '매출(만원)',          // 이름 변경
            position: 'left',
            axisLabel: {
                formatter: '{value}만'    // 단위 추가
            }
        },
        {
            type: 'value',
            name: '성장률(%)',           // 이름 변경
            position: 'right',
            axisLabel: {
                formatter: '{value}%'
            }
        }
    ];
    return response;
});
```

3. 단일 Y축으로 변경 (🟡 선택)

```
// 모든 데이터가 같은 단위라면
this.setYAxis = fx.curry((option, response) => {
    option.yAxis = {
        type: 'value',
        name: '금액(원)'
    };
    return response;
});

// loadChart에서도 변경
const loadChart = () => fx.go(
    this.getChartData(apiArguments),
    this.setXAxis(chartOption),
    this.setYAxis(chartOption),          // setDualYAxis → setYAxis
    this.setMixedSeries(chartOption),
    this.setTooltip(chartOption),
    (_) => this.settingChart(this.myMixedChart, chartOption)
);
```

4. Bar 스타일 커스터마이징 (🟡 선택)

```
this.setMixedSeries = fx.curry((option, response) => {
    option.series = response.series.map((serie, index) => ({
        name: serie.name,
        type: serie.type,
        data: serie.data,
        yAxisIndex: serie.yAxisIndex || 0,
        ...(serie.type === 'line' && {
            smooth: true,
            symbol: 'circle',
            symbolSize: 6,
            lineStyle: { width: 3 }      // 선 두께
        }),
        ...(serie.type === 'bar' && {
            barWidth: '30%',             // 막대 너비
            itemStyle: {
                borderRadius: [5, 5, 0, 0] // 상단 둥글게
```

```
        }
      })
    }));
    return response;
});
```

🔍 Preview 섹션 이해

Preview는 개발 중 미리보기용입니다:

```
const targetElement = this.element.querySelector('#echarts')
this.previewChart = echarts.init(targetElement);

// API와 동일한 구조의 샘플 데이터
const response = {
  "categories": [
    "Sep", "Oct", "Nov", "Dec", "Jan", "Feb",
    "Mar", "Apr", "May", "Jun", "Jul", "Aug"
  ],
  "series": [
    {
      "name": "Revenue",
      "type": "bar",
      "data": [2493, 3972, 4924, 2707, 2631, 2345, 3951, 2391, 1302, 3287, 4
726, 3827]
    },
    {
      "name": "Profit",
      "type": "bar",
      "data": [834, 955, 261, 705, 598, 952, 493, 885, 743, 461, 481, 351]
    },
    {
      "name": "Growth Rate",
      "type": "line",
      "yAxisIndex": 1,
      "data": [8, 4, -1, 28, 23, 13, 18, 20, 10, 11, -5, 12]
    }
  ]
```

```
};

// 실제 코드와 동일한 방식으로 처리
let option = {};

// X축
option.xAxis = {
  type: 'category',
  data: response.categories
};

// 이중 Y축
option.yAxis = [
  {
    type: 'value',
    name: 'Amount',
    position: 'left'
  },
  {
    type: 'value',
    name: 'Growth Rate (%)',
    position: 'right',
    axisLabel: {
      formatter: '{value}%'
    }
  }
];

// 혼합 시리즈
option.series = response.series.map(serie => ({
  name: serie.name,
  type: serie.type,
  data: serie.data,
  yAxisIndex: serie.yAxisIndex || 0,
  ...(serie.type === 'line' && {
    smooth: true,
    symbol: 'circle',
    symbolSize: 6
```

```
  }),
  ...(serie.type === 'bar' && {
    barWidth: '20%'
  })
}));

// 툴팁과 범례
option.tooltip = {
  trigger: 'axis',
  axisPointer: {
    type: 'cross'
  }
};

option.legend = {
  data: response.series.map(s => s.name),
  top: 10
};

this.previewChart.setOption(option);
```

Preview의 역할:

- API 연결 전에 차트가 잘 작동하는지 확인
- 실제 API 응답과 동일한 구조로 테스트
- 메인 코드의 처리 방식을 그대로 재현

⚠️ 자주 발생하는 문제

차트가 안 보일 때

- HTML에 `<div id="echarts"></div>` 확인
- div에 높이 지정: `<div id="echarts" style="height: 400px;"></div>`
- 브라우저 콘솔(F12)에서 에러 메시지 확인

데이터가 안 나올 때

- `datasetName`이 정확한지 확인

- **데이터 구조와 함수 매칭 확인:**
  - `setXAxis`에서 사용하는 `response.categories`가 있는지
  - `setMixedSeries`에서 사용하는 `response.series`가 있는지
  - 각 series에 `type`과 `data`가 있는지
  - 브라우저 콘솔에서 `console.log(response)`로 구조 파악

Bar가 겹칠 때

```
// barGap 조정
...(serie.type === 'bar' && {
  barWidth: '30%',
  barGap: '30%'  // 막대 사이 간격
})
```

Line이 Y축과 맞지 않을 때

- `yAxisIndex` 확인 (0 = 왼쪽, 1 = 오른쪽)
- Y축 범위 자동 조정 대신 수동 설정:

```
{
  type: 'value',
  name: 'Growth Rate (%)',
  min: -10,  // 최소값
  max: 30,   // 최대값
  position: 'right'
}
```

✅ 체크리스트

- ☐ HTML에 `<div id="echarts" style="height: 400px;"></div>` 존재
- ☐ `datasetName`을 내 API 이름으로 변경
- ☐ `params`를 필요한 값으로 수정
- ☐ **각 설정 함수가 올바른 데이터를 받고 있는지 확인**
  - X축 데이터 배열 확인

- Series 배열과 각 항목의 type 확인
- yAxisIndex가 필요한 경우 제대로 설정되었는지
- 필요시 함수 내부 수정

☐ destroy 탭에 `clearInterval(this.myInterval)` 위치 확인