47장 에러 처리

47.1 에러 처리의 필요성

- 에러가 발생하지 않는 코드를 작성하는 것은 불가능
- 발생한 에러에 대해 대처하지 않고 방치하면 프로그램은 강제 종료된다.
- Try catch 문을 사용하면 프로그램이 강제 종료되지 않고 계속해서 코드를 실행할 수 있다.

47.2 try ... catch ... finally 문

- 에러 처리 방법
  - if문이나 단축 평가 또는 옵셔널 체이닝 연산자를 이용해서 에러 처리
  - Try ..catch ... finally — Try { } catch (err) { } finally { }
- try코드 블록이 먼저 실행 — 에러가 발생하면 발생한 에러가 catch문의 err변수에 전달

예제

47.3 Error 객체

- Error 생성자 함수는 에러 객체를 생성
  - const error = new Error('invalid')
  - message 프로퍼티 — 생성자 함수 인수에 전달한 메세지
  - stack 프로퍼티 — 에러를 발생시킨 콜스택의 호출 정보를 나타내는 문자열 — 디버깅 목적으로 사용
- 생성자 함수 7가지
  - Error — 일반적인 에러 객체
  - SyntaxError — Js 문법에 맞지 않는 문을 해석할 때 발생
  - ReferenceError — 참조할 수 없는 식별자를 참조했을 때 발생
  - TypeError — 피연산자 또는 인수의 데이터 타입이 유효하지 않을 때 발생
  - RangeError — 숫자값의 허용 범위를 벗어났을 때 발생
  - URIError — encodeURI 또는 decodeURI 함수에 부적절한 인수를 전달했을 때 발생
  - EvalError — eval 함수에서 발생하는 에러

47.4 throw 문

- Error 생성자 함수로 에러 객체를 생성하고 throw 문으로 에러를 발생시킨다.
- Try 코드 블록에서 throw문으로 에러 객체를 던져야 함 — 던지면 catch문의 에러 변수가 생성되고 던져진 에러 객체가 할당
- throw new Error('error!');

47.5 에러의 전파

- 에러는 호출자 방향으로 전파된다. — 즉, 콜 스택의 아래 방향으로 전파

예제

- 주의할 점 — 비동기 함수인 setTimeout이나 프로미스 후속 처리 메서드의 콜백 함수는 호출자가 없다는 것
  - 이러한 것들은 태스크 큐나 마이크로태스크 큐에 일시 저장되었다가 콜 스택이 비면 이벤트 루프에 의해 콜 스택으로 푸시되어 실행
  - 콜 스택에 푸시된 콜백함수의 실행 컨텍스트는 콜 스택의 가장 하부에 존재 — 따라서 에러를 전파할 호출자가 존재하지 않는다.