SAMSUNG SW ACADEMY FOR YOUTH

對外秘

프로젝트 명세서

캐싱의 기본 개념 이해 및 적용

對外秘

목차

1. 프로젝트 개요

일반적인 어플리케이션을 구현함에 있어 사용자 요청에 대한 빠른 응답 속도는 매우 중요한 요소입니다. 어플리케이션 성능 측면에서의 중요 지표인 응답 속도를 줄이기 위한 방법 중 캐싱의 개념은 보편적으로 활용되고 있습니다. 우리가 평소 자주 사용하는 인터넷 웹 브라우저에서는 특정 사이트로부터의 이미지와 같은 리소스를 읽어 들이는 과정에서의 네트워크 전송 시간을 최소화하기 위해 캐싱을 적극 활용하고 있습니다. 또한 하드웨어 레벨에서도 이러한 캐싱 장치는 매우 중요합니다. 고성능 CPU 에는 고비용의 캐시 메모리가 포함되어 있다는 것에서 이런 내용을 확인할 수 있습니다.

그렇지만 일반적으로 캐싱은 데이터 원본을 보다 빠른 장소에 옮겨서 활용하는 만큼 보여지는 데이터가 원본의 자료와 항상 100% 일치할 수 없다는 단점을 가지고도 있습니다. 얻는 게 있으면 잃는 것도 있을 수밖에 없기에 우리는 캐싱을 활용함에 있어서 신중함을 기해야 합니다. 본 '자기 주도 프로젝트'에서는 캐싱의 개념을 알아보고 간단한 과제를 수행하면서 차후 마주하게 될 성능 지표의 첫 단추를 준비할 수 있는 기초 체력을 다져보고자 합니다.

2. 과제

1) 캐싱에 관한 기본 개념 정리

캐시는 왜 쓰는지? 그 기초 원리는 무엇인지? 핵심 개념을 이해해야만 캐싱을 프로젝트에 도입할 때에도 어떤 서비스에 캐시를 적용하는 것이 유용할지 또 위험 요소는 없는지 체크할 수 있는 능동적인 활용이 가능할 것입니다. 이에 우리는 이번 과제에서 1차적으로 캐싱에 관하여 기본 개념을 스스로 정리해보도록 합니다.

캐싱은 크게 어려운 개념이 아니며 자주 사용하는 대부분의 프로그램에 적용되어 있습니다. 예를 들어 브라우저는 페이지 요청 마다 네트워크를 통해 화면에 표시할 이미지를 가져오지 않습니다. 보통은 로컬 파일 저장 장치에 저장하여 두고 이를 바로 사용하게 됩니다. 이렇듯 상대적으로 느린 장치의 입출력을 통해 사용되는 정보를 보다 빠른 장치에 임시 저장하여 데이터로 활용함으로써 전체적인 성능 향상을 도모하는 메커니즘이 캐싱의 핵심입니다. 아래의 참고 링크뿐만 아니라 스스로 캐싱의 개념에 대해 설명할 수 있을 때까지 집중력을 가지고 학습하셨으면 합니다.

- 캐시 메모리
  https://namu.wiki/w/%EC%BA%90%EC%8B%9C%20%EB%A9%94%EB%AA%A8%EB%A6%AC

- 웹 어플리케이션 전반에 걸친 캐싱 정리
  https://opentutorials.org/course/697/3839

2) ‘SSAFY 카페’ 추천 메뉴 캐싱 서비스 구현

‘SSAFY 카페’는 매시간마다 변경되는 추천 메뉴를 ‘menu_hour.csv’ 파일에 기록합니다. 본 데이터 원본에 대한 읽기 작업이 상대적으로 느리게 작동하도록 구현하여 캐싱이 필요한 당위성을 부여해 주도록 합니다. 이후 해당 추천 메뉴 데이터를 원본 파일이 아닌 메모리로부터 읽어 올 수 있도록 캐싱을 도입하여

구현합니다. 프로그래밍 언어에는 제약이 없으며 구현 가능하고 내가 익숙한 언어를 선택하여 개발합니다.

a. 추천 메뉴 데이터 원본 저장소 구현 (파일 입/출력)

◆ 프로그램이 작동하는 현재 디렉토리내 'menu_hour.csv' 파일로부터 추천 메뉴를 읽어 콘솔에 출력합니다. 전체 메뉴는 "아메리카노, 카페라떼, 오렌지주스" 이고 이 중 한 가지를 파일에 기재하여 저장합니다.

◆ 캐싱의 당위성을 부여하기 위해 3초의 읽기 지연시간을 강제로 지정하여 개발합니다. (보통은 사용하는 언어에서 thread sleep 을 주는 기능을 활용합니다)

◆ 현재까지 작성한 프로그램을 실행하면 대략 3초 정도 후에 추천 메뉴가 콘솔 화면에 출력됩니다.

b. 처리 속도 개선을 위해 메모리에 원본 데이터 캐싱 (메모리 데이터 관리)

◆ 원본 데이터를 저장할 임시 장소를 메모리에 마련해두고 매 시간(정시)이 지난 시점 마다 파일로부터 읽어 들여 갱신할 수 있도록 구현합니다.

◆ 프로그램은 최초 실행 이후 강제 종료하기 이전에는 사용자의 키 입력 이벤트를 처리하기 위해 대기합니다. 키 입력 처리 이후 다시 대기 상태로 바뀌어 프로그램은 종료되지 않습니다.

◆ '1' 키가 입력되면 원본데이터로부터 '2' 는 메모리로부터 읽은 메뉴 정보를 콘솔에 출력합니다. 이 때 키 입력부터 출력까지의 응답시간을 함께 출력하도록 구현합니다.

c. 추천 메뉴 데이터 쓰기 구현 (파일 입/출력)

◆ 추천 메뉴를 변경하는 기능을 구현합니다. 현재 캐싱 임시 저장소와 원본

파일 저장소 두 곳이 존재하기에 쓰기 작업의 순서도 간과해서는 안 될 중요한 요소입니다. 실제 캐싱을 적용함에 있어 많은 고민의 포인트가 되는 지점이기에 신중하게 여러 경우의 수를 고려하도록 합니다.

◆ ‘3’ 을 입력한 이후 다시 한번 ‘메뉴명’ 을 입력하면 위의 추천 메뉴가 변경됩니다.

3. 심화 과제

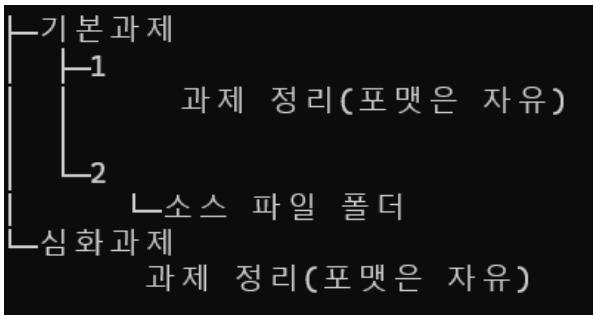
1) 기본 과제에서의 캐싱은 각각의 로컬 머신에 저장되는 '로컬 캐시'의 개념입니다. 캐시를 직접 구현하다 보면 캐시 만료 설정, 캐시 사이즈 등 고려해야 할 요소들이 굉장히 많으며 이런 기능들을 대거 제공해주는 훌륭한 '로컬 캐시 라이브러리'가 존재합니다. 내가 익숙한 언어 생태계에서 제공해주는 제품들에는 어떤 것들이 있는지 정리해봅시다.

2) 일반적으로 백엔드 서버는 싱글이 아닌 멀티로 구성합니다. 캐시 업데이트가 각각의 독립된 로컬 머신에서 수행되는 구조에서는 별도의 캐시 동기화 작업이 없다면 다른 서버 머신과의 데이터 불일치가 생길 수밖에 없습니다. 이런 측면을 고려할 때 '캐시 서버' 제품군은 우리에게 시사하는 바가 크다고 할 수 있습니다. 대표적인 제품인 'Redis'에 대해 알아보고 나만의 언어로 정리해봅시다.

4. 산출물 제출

- 제출 폴더는 https://lab.ssafy.com/s12-study/seasonal_fesw/ “산출물 제출 가이드.pdf” 의 내용을 참고하여 만들어주세요.
- 작성 내용

  아래의 트리처럼 과제 번호에 맞는 폴더에 문서를 첨부하고 전체를 압축하여 제출해주세요.



1) 캐싱에 관한 기본 개념 정리

  일반적인 캐시의 정의 및 장단점에 대해 학습하신 내용을 정리합니다. 분량 제한은 없으나 내가 이해한 내용을 정리하도록 합니다.

2) ‘SSAFY 카페’ 추천 메뉴 캐싱 서비스 구현

  구현이 가능하고 익숙한 언어로 과제 항목에 정의 된 요구 사항대로 서비스를 구현하고 해당 소스파일을 제출합니다.

3) 심화 과제

  로컬 캐시 라이브러리 제품들과 캐시 서버 솔루션 ‘redis’에 대해 필요성과 장단점에 대해 충분히 고민하고 해당 내용을 정리합니다.

5. 추천 학습 방법

1) 캐시 본연의 의미를 파악한 이후 캐시 적용 시나리오를 능동적으로 구성해보는 시간을 갖는 게 1차적인 목표입니다. 파일 입출력을 비롯한 몇 가지 개발 요소들은 스스로 책, 블로그, 영상 컨텐츠 등을 통해 어렵지 않게 정복해 나가시길 바랍니다.

2) 1)번 이후 내가 선택한 프로그래밍 언어, 웹 프레임워크에서 많이 사용되는 캐시 솔루션(라이브러리, 제품)에는 어떠한 것들이 있는지 'redis'를 포함하여 조사해보고 각각의 장단점을 파악해 봅니다. 가까운 미래에 개발 기술 스택에 포함될 가능성이 매우 크기에 이를 고려하여 정리해 보길 바랍니다.