|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **주차별 학습일지**      **2019년 2 학기**  **학습기간 : 2019년 12월 9일 ~ 2019년 12월 13일**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **학 과** | **융합소프트웨어과** | | **학년-반** | **1 - 1** | | **학 번** | **201953204** | | **이 름** | **이근혁** | | **교과목** | **캡스톤디자인1(PBL)** | | | | | **점 수** |  | **교과목담당교수 : 류중경 교수님**  **이영걸 교수님** | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 학습기간 | 2019년 12월 9일 ∼ 2019년 12월 13일 | | 학습근로자명 | | 이근혁 |
| 교과목 | 캡스톤디자인1(PBL) | 기업현장교사 | | 오 태 현 (인) | |
| 대학지도교수 | | 류 중 경 (인) | |
| 능력단위명 | 캡스톤디자인1(PBL) |
| 대학지도교수 | | 이 영 걸 (인) | |
| 과제명 | **서비스워커 개발** | | | | |
| |  | | --- | | 붙임 1) 주차별 학습일지(학생용) |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | **학 과** | **융합소프트웨어과** | **학 번** | **201953204** | **성 명** | **이근혁** | | **회사명** | **(주)아임클라우드** | | | | | | **교과목** | 캡스톤디자인1(PBL) | | | | | | **내 용 (아래 항목을 참조하여 자유로이 기술)** | | | | | | | **1. 학습목표**  개발 계획서에 따라 요구되는 기능을 구현할 수 있다. 웹 페이지 기능 외에 푸시, 오프라인 캐싱을  위한 서비스워커 구현을 목적으로 진행한다.  **2. 학습내용**  지난 주차에 웹 페이지 기능 개발을 마무리하여 오프라인 동작을 위한 캐싱과 푸시 알림을 수신하기  위해 서비스워커를 개발했다. 서비스워커는 설치 -> 활성화 순서로 작동하게 되며 HTTPS 혹은  로컬호스트에서만 설치되고 동작되는 스크립트이다. 브라우저에서 동작하긴 하지만 UI 스레드(메인)  이 아닌 별도의 스레드에서 동작하므로 직접적인 DOM 접근이 불가능하다. 하지만 UI 스레드와  메시지를 통해 데이터를 주고받을 수 있으므로 어느정도 보완할 수 있다. 최종적으로 오프라인 캐싱  과 푸시알림 이벤트를 받기 위한 리스너를 구현하여 마무리했다.  **3. 관련자료**  (모든 문서와 개발 자료는 깃허브([https://github.com/leegeunhyeok/noty](http://github.com/leegeunhyeok/noty))에 기록 및 관리 중)  스크린샷이(가) 표시된 사진  자동 생성된 설명  (서비스워커의 전체적인 생명주기 – 설치 이후 활성화가 되어야 Fetch, push등의 이벤트를 제어할  수 있다.)  (태그 불러오는 코드 – GraphQL 스키마에 맞게 쿼리를 작성하여 쉽게 데이터를 불러올 수 있었다.)  스크린샷이(가) 표시된 사진  자동 생성된 설명스크린샷, 모니터이(가) 표시된 사진  자동 생성된 설명  (완성된 노트, 태그 등 UI 및 기능 – 데이터를 추가, 삭제, 수정 가능하며 GraphQL를 통해 서버로  전달되어 데이터가 처리된다.)  스크린샷, 모니터, 화면, 전화이(가) 표시된 사진  자동 생성된 설명  (설치되어 정상적으로 활성화된 서비스워커 모습)  스크린샷, 모니터, 화면이(가) 표시된 사진  자동 생성된 설명  (캐시 스토리지에 저장된 리소스 목록 – 캐시되어있기 때문에 기기의 네트워크가 불안정하거나 끊겨  있어도 네이티브 앱처럼 웹 페이지를 정상적으로 사용할 수 있다!)  **4. 학습성과 및 기대효과**  오프라인 동작을 위한 서비스워커를 구현해보았다. 별도의 스레드에서 동작하는 만큼 생소한  개념이 많았지만 한 번 개념을 잡고 개발하니 큰 어려움은 없었다. 다만 웹 분야의 신기술인 만큼  구형 브라우저에선 지원하지 않는 경우가 많아 실제 기업에서 상용 서비스로 개발하기에는 조금  제약사항이 많을 것 같다. 하지만 서비스워커를 지원하지 않는다면 기존처럼 오프라인에서 동작하지  않는 웹 페이지는 그대로 사용할 수 있으며 서비스워커를 지원한다면 사용자도 눈치채지 못하도록  “점진적”으로 개선된 웹 기능을 사용할 수 있다는 장점이 있다. 앞으로 다양한 기술을 여러 브라우저  에서 접할 수 있도록 많은 개발자들이 노력하고 있으니 꾸준히 노력하는 자세를 지녀야겠다. | | | | | | | | | | | |