2주차

리처드 스톨만 - GNU 프로젝트와 자유 소프트웨어 재단 설립자 Copyleft 의 개념, GPL 라이선스 개념 도입
GDB, Emacs, GNU 컴파일러 등

Free software foundation

1. 프로그램을 어떤 목적을 위해서도 실행할 수 있는 자유
2. 프로그램의 작동 원리 연구하고, 이를 자신의 필요에 맞게 변경시킬 수 있는 자유 -> 소스코드 접근 선행
3. 이웃을 돕기 위해 프로그램을 복제하고 배포할 수 있는 자유
4. 프로그램 향상시키고 이를 공동체 전체의 이익을 위해 다시 환원시킬 수 있는 자유

GNU

gzip, gimp, gcc, bash, openoffice 등

오픈소스 10개 법칙?

1. 무료 재배포
    - 라이센스는 로열티 또는 판매비용 없다
2. 소스코드
    - 프로그램에는 소스코드가 포함
3. 파생 작업
    - 라이센스는 변경 및 파생작업이 가능해야 한다. 원래 소프트웨어 라이센스와 같은 조건 하에 배포
4. 소스 코드의 무결성
    - 프로그램 변경 목적으로 패치 파일 배포할 경우 라이센스 제한. 변경된 소스 코드에서 구현된 소프트웨어 배포 허용해야함
5. 개인 또는 그룹의 평등
    - 라이센스는 어떤 개인이나 그룹에 차별을 둬서는 안된다.
6. 분야에 대한 평등
    - 라이센스는 특정 분야에서 프로그램을 사용하는 것에 대해 제한을 둬서는 안된다.
7. 라이센스 배포
    - 프로그램에 대한 권한은 프로그램이 재배포된 모든 곳에 적용되어야 한다. 추가 라이센스를 발행할 필요가 없다
8. 제품 스팩에 따른 라이센스
    - 프로그램이 그 배포판에서 추출되었고, 그 프로그램의 라이센스 조건 하에 사용 및 배포된다면 재배포된 프로그램을 사용하는 모든 당사자들은 원래의 소프트웨어 배포판에 허용된 것과 같은 권한을 가지게 된다.
9. 라이센스는 다른 소프트웨어를 제한하지 않는다.
    - 라이센스를 받은 소프트웨어와 함께 배포된 다른 소프트웨어에 제약 사항을 둬서는 안된다
10. 라이센스는 기술 중립적이어야 한다
    - 라이센스는 기술이나 인터페이스 스타일을 한정해서는 안된다.

오픈 소스는

프로그램의 최종 결과물만 공개하는 것이 아니라 작업의 중간과정과 소스코드를 함께 공개/공유 하는 것 그리고 그 결과를 함께 나눔으로써 세상을 더 풍요롭게 만들어 주는 것.

구글 오픈소스 프로젝트

- android
- chromium
- angularjs
- go
- gwt
- dart

네이버 오픈소스 프로젝트

- egjs
- pinpoint
- lucyxss
- arcus
- xpress engine
- cubrid
- ngrinder

초보 개발자가 오픈소스에 기여하는 5단계

1. 관심분야 선택하기
2. 커뮤니티 찾기
3. 문서 접하기
4. 써보기
5. 깃허브 배우기

그 외

- 첫 목표는 언제나 쓸만한 무언가를 만드는 것
- 다음은 계속 개발하고 공유하는 것
- 마지막으로 기여자를 받아 들이는 것

1. 최적의 시장 진입 타이밍, 적당한 때에 문제 해결하는데 필요한 제품 제공
2. 개발자와 비개발자가 포함된 능력있는 팀
3. 참여 설계. 이 리스트에 언급되는 것보다 훨씬 구체적인, 코드와 코드의 구조를 포함하고, 그 너머까지 포함하는 설계
4. 모듈화가 잘 되어 있는 코드, 기여자가 한번에 어느 기능이 어떤 파일의 어느 부분에 있는지 알 수 있어야 한다.
5. 넓게 적용할 수 있는 코드 또는 니치한 필요보다 좀 더 많은 사람에게 도달가능한 코드
6. 훌륭한 최초 코드
7. 관대한 라이센스

오픈 소스 프로젝트에 참여하면?

1. 학교에서 배운 이론을 실제로 적용해봄

2. 다양한 사람들과 협업하는 경험
3. 고수가 작성한 코드를 읽을수 있음
4. 다른 사람이 내 코드 리뷰해줌
5. 나도 다른 사람의 코드를 읽으며 리뷰
6. 학교에서 아무도 신경쓰지 않는 것들의 중요성을 깨닮음
7. 커뮤니케이션, 문서화, 테스트 케이스 등
8. 취직할 수 잇음

2주차(2)

저작권이란?

- 인간의 사상 또는 감정을 표현한 창작물인 저작물에 대한 배타적, 독점적 권리이다.
- 저작권 표시가 없어도 저작권법에 의해 보호받는다.
- 창작한 순간 자동적으로 발생

오픈소스 라이선스

- 오픈소스 SW 저작권자가 자신의SW에 대해 사용 조건 및 범위를 명시한 계약

라이센스 확인해야 하는 것

1. Permissions
    - Commercial use
    - Distribution
    - Modification
    - Patent Use
    - Private Use
2. Limitation
    - Liability
    - Trademark use
    - Warranty
3. Conditions
    - Disclose source
    - License and copyright notice
    - Network use id distribution
    - Same license
    - State changes

라이선스별 특징

1. MIT
    - 저작권 표시
    - 라이선스 표시
2. BSD 3-Cluase
    - 저작권 표시
    - 라이선스 표시

3. Apache 2.0
    - 저작권 표시
    - 라이선스 표시
    - 수정내용 표시
4. MPL 2.0
    - 저작권 표시
    - 라이선스 표시
    - 라이선스 승계
    - 소스 공개
5. LGPL v2.1
    - 저작권 표시
    - 라이선스 표시
    - 수정내용 표시
    - 라이선스 승계
    - 소스공개
6. GPL 2.0
    - 저작권 표시
    - 라이선스 표시
    - 수정내용 표시
    - 라이선스 승계
    - 소스 공개

라이선스 확인법

- README에 포함되거나
- LICNESE 파일에 찾아봄
- 홈페이지가 있는 경우 - License 관련 메뉴, SW설명 등
- 소스 코드내 주석

1. 라이선스를 확인하는 습관
2. 원저작자의 저작권과 라이선스 문구는 항상 유지
3. 다른 사람의 소스 코드를 가져올 때는 항상 출처 명시

3주차 - Chromium

- 구글 크로미움은 멀티 프로세스로 되어 있다
- 쓰레드와 프로세스의 차이점은 메모리 공유의 차이
- 멀티 쓰레드로 구현할 경우 한 탭에서 다른 탭 들간의 메모리가 공유되어 있기 때문에 보안적인 이슈가 있을 수 잇음-> 프로세스로 분리해서 원천적으로 차단함
- 크롬 프로세스 매니저 프로세스와, 각 탭 마다 프로세스가 있음
- 1 Browser Process, several renderer processes

클라이언트와 서버

1. 클라이언트 - 서버는 소켓으로 연결됨
    - 일반적으로 소켓 통신은 stream으로 되어 있다
2. 웹서버는 HTTP Request 와 Response 각 한번 씩만 하고 연결을 끊는다

크로미움 internal

Loading - Parsing - Layout - JS Execution - Painting - UIUpdate 과정을 거침

1. Parsing

- HTML Parsing -> DOM Tree 생성
- CSS Parsing - CSSOM Tree 생성
- DOM Tree + CSSOM Tree => Render Tree(Paint Tree)

2. Layouting : DOM 위치의 정의

- Layouting은 언제 일어 나는가
    - 첫번째 DOM 생성시
    - HTML Chunk가 발생할 때 by script
    - HTML Element 크기나 위치가 변경될 때
    - Sub-resources(e.g images)가 로드 될 때
- Render Tree를 돌면서 Paint!
- 여기서 문제점
    - RenderObject 의 층을 모른다
    - Layoring
    - Layor 생성 조건
        - 루트 오브젝트인 경우
        - CSS position
        - 투명한 경우
        - overflow, alpha mask, relfection을 가지는 경우
        - CSS filter를 사용하는 경우
        - GPU가속을 사용하는 canvas DOM
        - video 태그

요약

- html을 로드 -> Parsing -> DOM Tree 생성
- CSS 로드 -> parsing -> CSSOM Tree 생성
- DOM tree + CSSOM Tree -> Render Tree 생성
- Render Tree 에 대한 layouting 수행
- Paint Order을 결정하기 위해 Layering 수행 -> Render Layer tree 생성
- Graphics Library 사용 -> Paint

크로미움 internal 3

1. Javascript Engine으로 V8 Engine 사용
2. UI Engine으로 Blink 사용

V8 binding

1. V8에서 js를 해석한다. 이때 v8은 DOM트리를 모르기에 blink 엔진과 v8 엔진을 연결하는 v8 binding을 이용하여 blink엔진이 이를 실행하게 한다
2. v8 binding 은 type checking&converting 수행
3. 사용자가 직접 짜지않고 WebIDL으로 부터 자동 생성
4. WebIDL은 웹 표준에 의해 정의된다.

- crbug.com
    - for : hotlist=goodfirstbug 검색 -> 비교적 쉬운 버그
- cs.chromium.org
- https://github.com/romandev/chromium101
- https://www.chromium.org/Home
- Tests in Chromium
    - layout tests
    - browser tests
    - unit tests
    - gpu tests
- Code Review
    - Chromium 은 Gerrit이라는 사이트를 사용함

5주차 - Python

- 굉장히 다양한 분야에서 사용됨
- 사용자는 성별뿐만 아니라 모든 것에 대해 다양함
- CG - Maya Blender (3D 그래픽 소프트웨어), Disney ILM
- Music - Ableton Live

스프린트

- 스프린트는 관심있는 오픈 소스 프로젝트를 같은 장소에 모여 집중적으로 개발하는 자리
- 오픈소스 지식과 경험을 나눔
- 함께 면대면으로 만나 4일동안 intensive learning
- Wordwide pandas sprint
- DjangoCon sprint
- PyData sprint
- mozilla doc sprint
- 공개sw개발자센터 컨트리뷰톤
- 2002년 Cpython/Zope 커뮤니티에서 시작
- 2003년 pycon과 함께 시작
- 2006- 4days of sprint
- 2008 - 20+ 프로젝트/250명 참가자

- 2016 - 한국 pycon 개발 스프린트 시작

파이썬 커뮤니티의 원동력

- 동료의식에서 시작되는 존중과 배려
- 를 바탕으로 한 소속감과 참여감
- 에서 비롯되는 기여
- 를 통해 발전하는 생태계
- 에 모여드는 사람들
- 사이에서 싹트는 동료 의식
- (반복)

6주차 - react

1. HTML이란
    - 웹 페이지상에서 문단,제목,표,이미지,동영상 등을 정의하고 그 구조와 의미를 부여하는 마크업 언어
2. CSS란
    - 배경색,폰트,컨텐츠 레이아웃등을 지정해 html 컨텐츠를 꾸며주는 스타일 규칙 언어이다
3. Javascript
    - 동적으로 컨텐츠를 바꾸고,멀티미디어를 다루고,움직이는 이미지등 웹페이지를 꾸며주도록 하는 프로그래밍 언어이다.
    - 웹 뿐만 아니라 서버,응용 프로그램, 모바일 앱 모두 사용 가능
    - 많은 사용자와 방대한 자료
    - 계속 발전하는 언어
4. DOM?
    - 문서객체 모델
    - HTML,XML 문서의 프로그래밍 인터페이스
    - 문서의 구조화된 표현을 제공하며 프로그래밍 언어가 DOM구조에 접근 할 수 있는 방법을 제공한다.

React

- UI를 위한 자바스크립트 라이브러리
- 모바일 앱, 웹 사용 가능
- Facebook 내부 개발
- 2013 오픈소스 런칭
- 웹에서 UI 처리 부분에 있어 특화
- 컴포넌트 기반, 재사용성 추구
- JSX,ES6 문법 권장 -> babel 을 이용해 ES5로 변환
- 단방향 데이터 흐름
    - props(properties) 를 통해 데이터 전달
    - 리엑트에서 추구하는 컴포넌트 전달 방식
    - 부모가 자식에게 일방적으로 전달
    - 재사용성 높임
- ReactDOM - Virtual Dom을 이용해 퍼포먼스 극대화

    - 엘리멘트와 자식들을 전의 것과 비교해 필요한 부분만 업데이트 하도록 DOM에 요청한다
    - 기존 DOM들의 단점을 해결한다
- React Component
    - 재사용 가능한 독립적인 UI 구성 단위
- 장점
    - 체계적이고 유지 보수가 편리한 앱 제작 가능
    - 큰 규모에서도 퍼포먼스가 좋다
    - Web과 Native 앱 개발에 모두 사용 가능하다

React Native

- React의 개념 그대로 모바일앱에 적용
- 하이브리드 앱과는 다르다
- 자바스크립트로 iOS,Android 개발 가능하다
- 네이티브 앱과 비슷한 퍼포먼스를 보인다
- Native Modules과 JS Virtual Machine을 이어주는 RN Bridge가 있음

오픈 소스 하는 법

- Git
- Github
- StackOverflow

결론

- 자바스크립트 짱짱
    - 프론트 백 웹 앱 데스크탑 다!
- 거대한 커뮤니티, 많은 공급 수요

7주차 - js ,vue.js

- 오픈소스가 우리에게 주는 이점
    - 기술 발전
    - 산업 성장
    - 공유하고 나누는 문화
    - 빨라진 퇴근
- 오픈소스란 내 삶은 윤택하게 하는 집단 지성

Vue.js

- 작은 화면단 라이브러리 역할부터 큰 규모의 웹 애플리케이션 개발을 돕는 프레임워크 역할까지 점진적으로 적용할 수 있는 프론트앤드 프레임워크
- 컴포넌트 기반 개발 방식
    - 화면을 여러 개의 작은 단위로 쪼개어 개발
    - 재사용성, 구현속도, 코드가독성 증가
- MVVM 패턴
    - 화면 UI코드와 백앤드 데이터 처리 코드를 분리

- 앵귤러와 리액트의 장점 흡수
    - Two way data binding
    - Virtual DOM
- 뷰 인스턴스
    - 뷰로 화면 개발할때 필수로 생성해야 하는 단위
    - new Vue({})~~
- 뷰 컴포넌트
    - 화면을 구조적으로 설계하기 위한 요소
    - Vue.component('my-cmp', {})~
- 뷰 라우터
    - 여러 개의 화면 간에 이동하는 방법
- 뷰 템플릿
    - 화면을 구체적으로 꾸미는 방법 & 문법
    - html에 코드를 씀