|  |
| --- |
| 오픈소스 SW 프로젝트 team9 final report |
| 조원: 김민주, 석제노, 유현욱, 조성찬 |



Contents

1. 프로젝트 소개

2. 관련 기술 소개

3. 구현물에 대한 설명

- MVP 기능 설명

- 현재 프로그램의 usage scenario

- 프로그램 설치 방법

- 핵심 기술에 대한 설명

4. 발표 이후 개선된 부분

5. 최종 구현물 데모

6. GitHub Repository 내역

**1. 프로젝트 소개**



QRdoc은 문서 안 링크 주소를 QR-code로 변환하여 문서에 넣어주는 프로그램입니다.

1. QRdoc을 사용하면, 문서 안 모든 링크를 일일이 QR-code로 변환하지 않아도 됩니다.

**클릭 한 번으로 문서 안 모든 링크 주소를 QR-code로 만들어줍니다.**

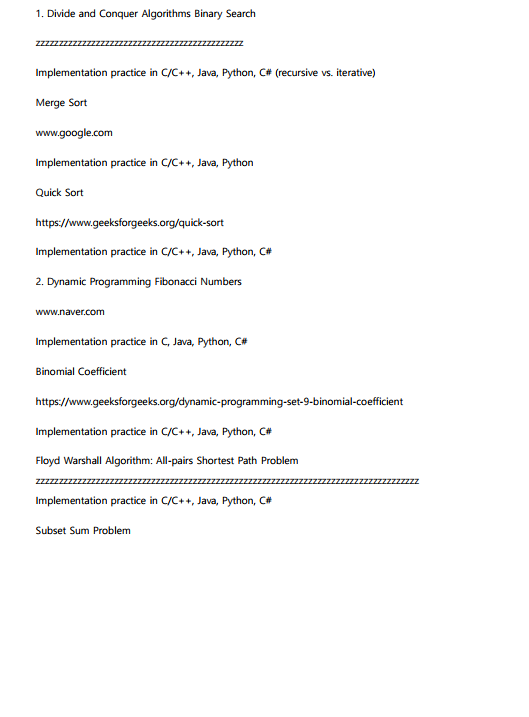
1. QRdoc을 사용하면, 문서를 읽는 사람이 링크주소를 직접 인터넷 주소창에 타이핑하지 않아도 됩니다.

**QR-code로 손쉽게 해당 사이트에 접속할 수 있습니다.**

1. QRdoc을 사용하면, 여러 pdf 문서를 한 번에 관리할 수 있습니다.

**클릭 한 번으로 여러 pdf 파일 안에 QR-code를 첨부할 수 있습니다.**

ex)



**2. 관련 기술 소개**

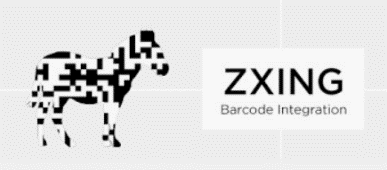
사용한 라이브러리:

1. **PDFBox**: A library to **read** and **write PDF** in Java.

https://pdfbox.apache.org/



1. **ZXING**: A library to **encode** and decode **QR-code** in Java.

****https://github.com/zxing/zxing

1. Image4j: A library to read and write BMP, **ICO** image formats in Java.

http://image4j.sourceforge.net/



1. Javafx: standard **GUI** library for Java SE

https://openjfx.io/

1. Virtual Printer: A library to easily create a **windows virtual printer**.

https://github.com/TheHeadlessSourceMan/virtualPrinter



1. Ghostscript: An interpreter for the PostScript language and for **PDF**.

https://www.ghostscript.com/

주요하게 사용하는 라이브러리: **PDFBox, ZXING**.

**PDFbox**와 **ZXing**를 활용하는 프로그램은 많이 있지만, 별다른 조작 없이 문서를 넣으면 자동으로 오프라인 출력물에 적합하게 편집해 주는 프로그램은 지금껏 없었습니다. PDF파일을 생성하거나, 폰트 및 품질을 조정하는 프로그램 또는 QR-code를 생성, 인식하는데 초점이 맞춰져 있는 프로그램들만 있을 뿐입니다.

저희 프로젝트는 문서를 만들 때, 문서 안 정보에 사람들이 접근하기 쉽도록 문서를 수정하는 것이 주 목적입니다. 문서 안 링크 주소를 QR-code로 변환하여 사용자에게 편의를 제공할 수 있습니다

또한 **가상프린터** 기능을 활용하여, 사용자가 기존 문서를 손쉽게 QR-code가 들어간 PDF 문서로 변환할 수 있게 합니다.

출력 버튼을 누르는 것만으로 출력물의 오프라인에서의 접근성을 높여주는 프로그램, 어디에도 없는 저희 프로젝트만의 차별성입니다.

1. **구현물에 대한 설명**

MVP (minimum viable product)의 기능

* 정규식 을 사용하여, PDF 문서에서 링크 주소를 추출한다.
* 링크를 **QR-code**로 변환한다. (ZXING 라이브러리 사용)
* PDF 문서에 QR-code를 삽입한다. (PDFBox 라이브러리 사용)

추가 기능

Favicon: 

* **favicon**을 QR-code의 배경으로 하는 **Visual QR-code**를 만든다.

ex) <https://github.com/OOS-Team-9/QRdoc> => [](https://github.com/OOS-Team-9/QRdoc/blob/master/documents/README_image/Visual_QR.png)

* **가상 프린터를** 사용해서, 한글 파일 또는 워드 파일을 pdf로 PDF 파일로 바꿔준다.

아직 구현하지 못한 기능

* PDF 문서 안 **하이퍼링크**를 찾아서, 그에 맞는 QR-code를 문서에 넣어준다.

#### 현재 프로그램의usage scenario

* **게시판에 붙일 문서 출력하는 경우**

1. 게시판에 붙여 놓을 문서를 작성한다.
2. 해당 문서는 링크가 들어가 있어서, 문서를 읽는 사람이 일일이 링크를 주소창에 타이핑해야 한다.
3. 그럴 때는 QR-doc 프로그램을 사용한다.
4. 문서 출력 버튼을 누른 다음, 예전에 설치한 QR-doc 가상 프린터를 설치한다.
5. 그러면 QR-doc 프로그램이 자동으로 문서 안 링크를 추출하고, 여백을 확인하여 QR-code를 삽입해준다.
6. 출력된 문서를 게시판에 붙인다.

* **기존에 작성한 문서 수정하는 경우**

1. 예전에 작성한 PDF 문서가 여러 개 있다.
2. 이 문서를 열어서 일일이 링크가 있는지 확인한 다음, QR-code를 만들어서 PDF문서 안에 삽입하기 귀찮다.
3. 이전에 다운로드 받은 QR-doc 프로그램을 실행한다.
4. 해당 PDF문서를 선택한다.
5. 변환 버튼을 누른다.
6. GUI창으로 변환 결과를 확인한다.
7. 변환 결과가 만족스러우면 결과 파일을 저장한다.

* **QR-code가 어떤 링크로 연결되는지 한눈에 확인하고 싶은 경우**

1. 링크를 QR-code로 변환하여 PDF문서 안에 넣고 싶다.
2. 그런데 이 QR-code가 정확히 어떤 링크로 연결되는지 사용자가 한눈에 확인할 수 있게 만들고 싶다.
3. 이전에 다운로드 받은 QR-doc 프로그램을 실행한다.
4. [](https://github.com/OOS-Team-9/QRdoc/blob/master/documents/README_image/Visual_QR.png)해당 문서를 선택한다.
5. 변환 버튼을 누른다.
6. <https://github.com/OOS-Team-9/QRdoc> =>
7. 이 QR-code가 github 링크로 연결된다는 사실을 사용자는 한눈에 알 수 있다.

#### 프로그램 설치 방법

1. 가상 프린터 설치

1-1. [HP Color LaserJet 2800﻿ 통신 드라이버 다운로드 링크](https://support.hp.com/kr-ko/drivers/selfservice/closure/hp-color-laserjet-2800-all-in-one-printer-series/445194?ssfFlag=true&sku=) 클릭

-> 설치 파일 다운로드 후 실행

1-2. virtual printer 설치 파일 다운로드. 압축 파일 풀기

1-3. virtual printer 폴더 안 gs952w32.exe 실행.

1-4 python-2.7.18.amd64 (1).msi 실행

1-5 python-2.7.18 환경 변수 추가 ([참고](https://wxmin.tistory.com/121))

1-6 \install printer driver\create printer 실행

1-7. \printer server\start printer server 실행

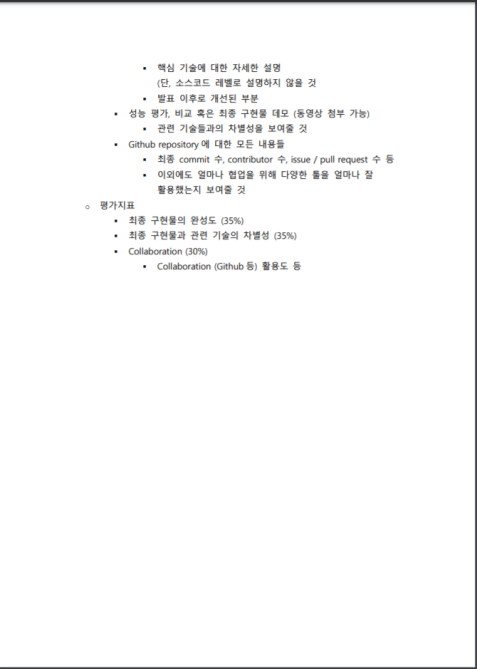
\*나중에 다시 실행할 때는 \printer server\start printer server만 실행

2. 프로그램 실행

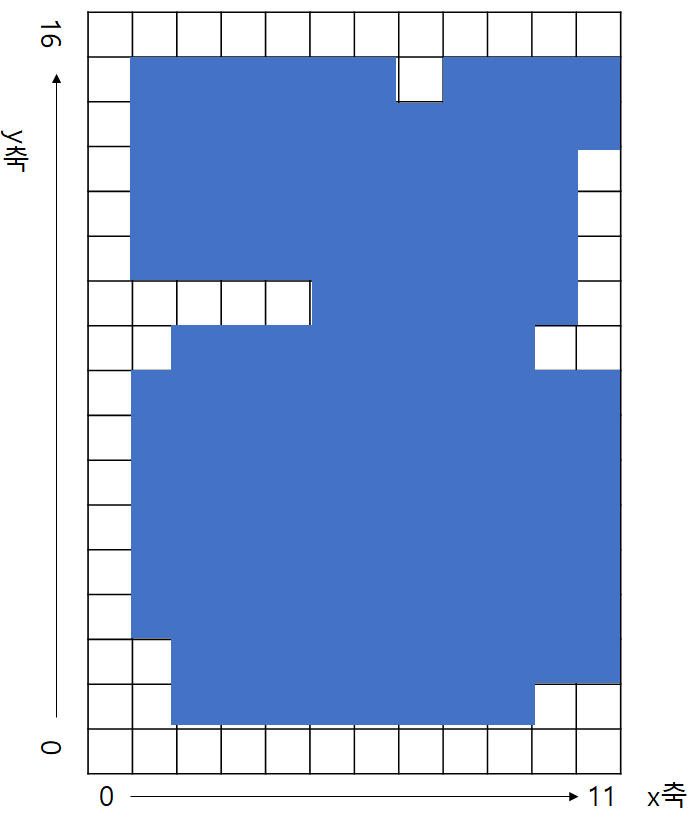
2-1. \QRdoc\out\artifacts\QRdoc\_jar 안에서 " java --module-path javafx-sdk-11.0.2\lib --add-modules javafx.controls,javafx.fxml --add-exports javafx.graphics/com.sun.javafx.sg.prism=ALL-UNNAMED -jar QRdoc\_v.jar" 명령어 입력.

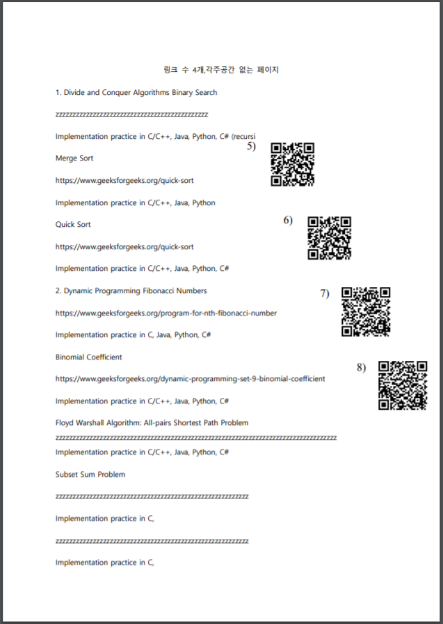
#### 핵심 기술에 대한 설명

1. **문서 내의 공간(여백)인식**



문서 내에 텍스트, 이미지 확인을 통해 여백을 확인합니다



페이지를 16\*11의 격자로 나누어 해당 공간이 여백인지 확인 후 True/False로 반환합니다.

확보된 여백에 QR-code들을 전부 삽입할 수 있다면 링크에 가장 가까운 여백에 생성된 QR-code들을 삽입합니다.

1. **Visual QR-code** 생성
2. Link extractor 로 문서에서 link를 추출하여 QR-code를 만듭니다.
3. link URL에서 host URL을 추출합니다.
4. host URL 끝에 “/favicon.ico”를 붙여 favicon.ico 파일을 다운로드 합니다.
5. QR-code, favicon, domain name. 이 세가지를 한데 모아 하나의 QR-code를 만듭니다.
6. 사용자는 visual QR-code를 통해서 변환된 QR-code가 어떤 주소의 것인지 간략하게 알 수 있습니다.

**URL**



**host**

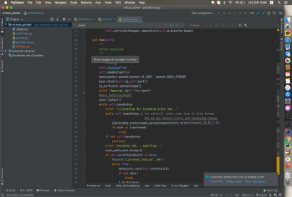
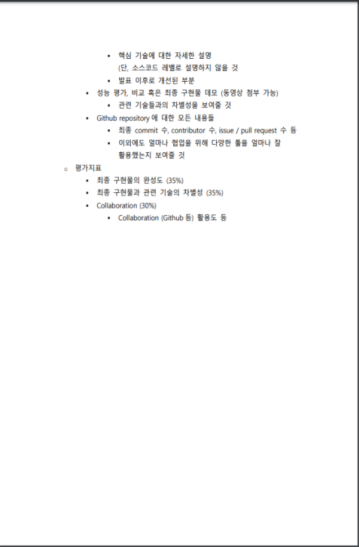


**favicon**

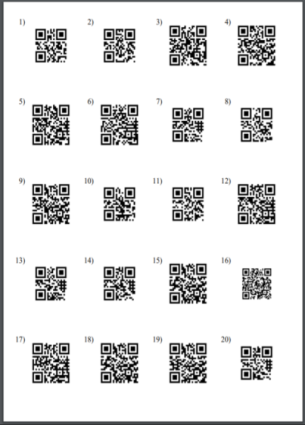
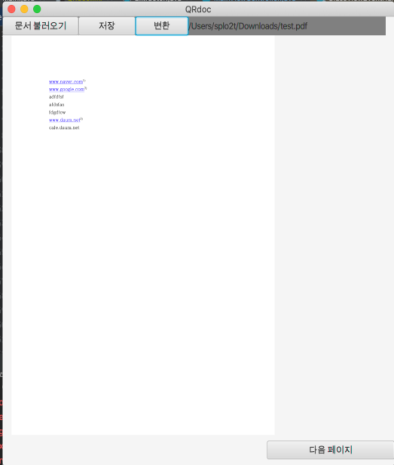


1. **가상프린터**
2. 파이썬에서 윈도우 스크립트를 통해 윈도우에 내장된 특정 프린터 드라이버를 설치합니다.
3. 설치한 프린터를 통해 인쇄하면 파이썬에서 TCP/IP통신을 통해 데이터를 불러옵니다
4. 불러온 데이터를 ghostscript 라이브러리를 통해 파싱해서 pdf로 변환합니다.
5. 변환한 파일을 프로그램으로 넘겨 QR-code를 생성, 삽입합니다.

.



TCP/IP



프린터 드라이버

1. 발표 이후 개선된 부분

발표 중 시연된 데모 영상에서 링크에 가장 가까운 여백에 삽입 되어진 QR-code들이 어색하다는 의견을 반영하여, 링크의 textPosition 기준 우측 하단에 위치한 여백을 최우선으로 확인하여 QR-code를 삽입하게끔 수정하였습니다.

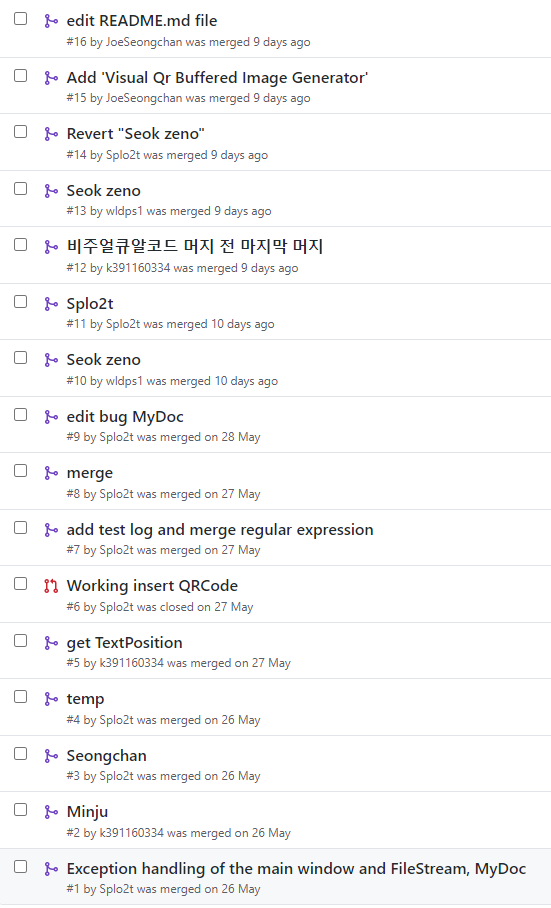
여백의 수가 많지 않을 경우, 데모영상에서 보여주었던 것처럼 링크 상단의 여백에도 삽입이 될지, 링크의 우측 하단에 위치한 여백만 인식하여 부족하면 미주로서 삽입이 될지는 유저 UI 추가를 통해 사용자의 선택에 맞기거나 하는 등의 더 좋은 방안을 찾고 있습니다.

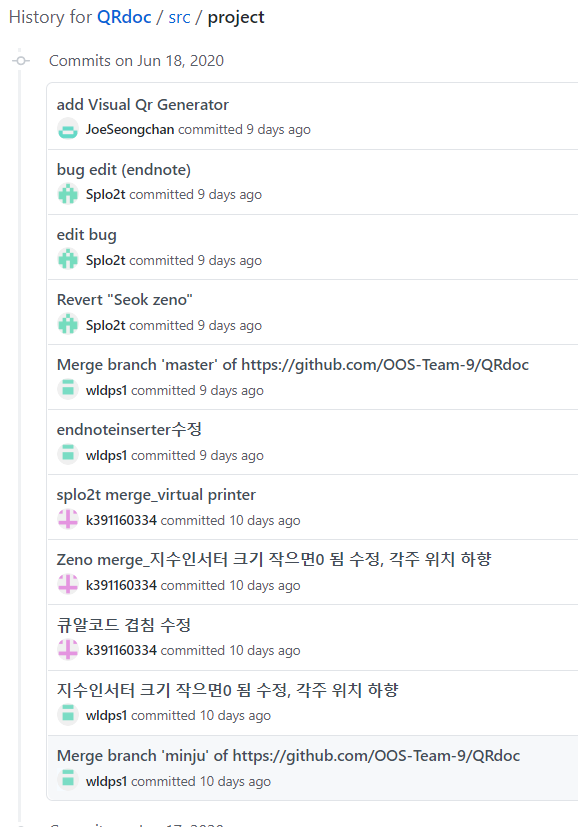
**4. 최종 구현물 데모**

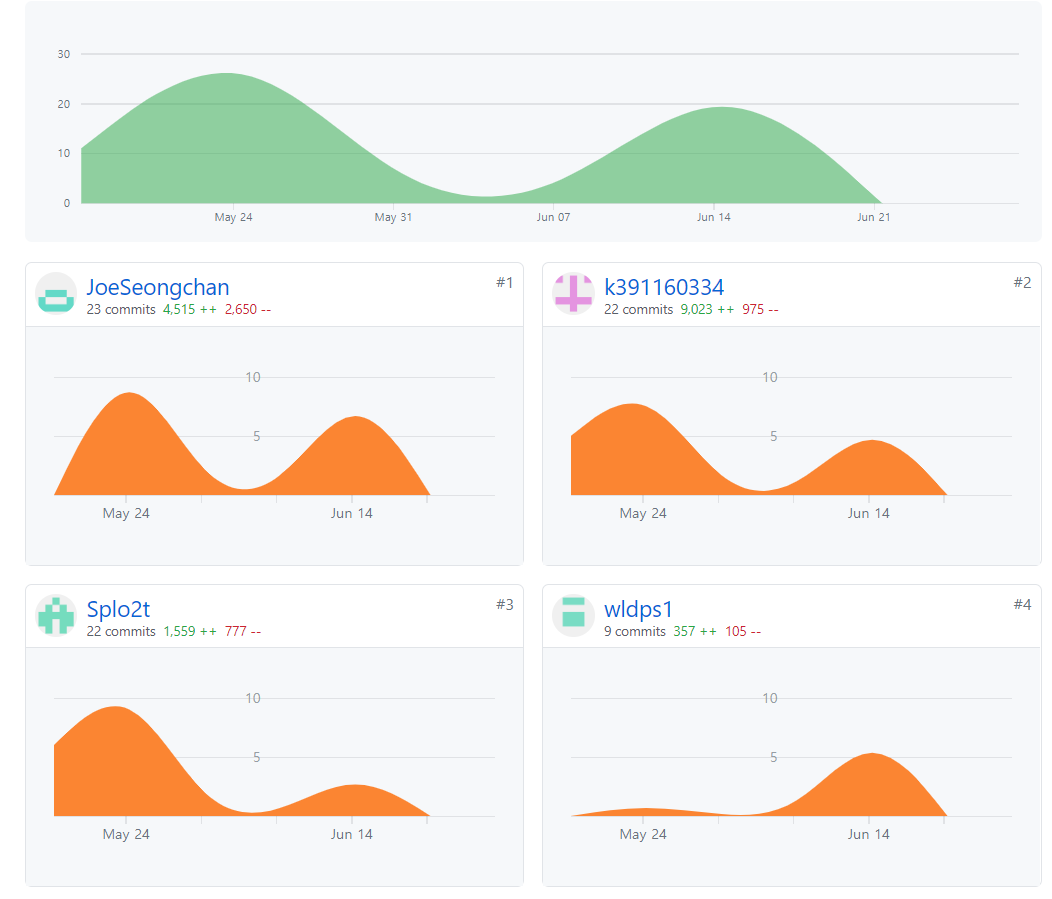
본 문서와 함께 실행파일, 가상프린터,결과물 캡처파일,

발표 때와 다른 데모영상 같이 첨부하여 붙입니다

**5. GitHub Repository 내역**

History/pull request 내역





Contributor insights

조성찬/김민주

조성찬

유현욱/석제노

유현욱