hci lab
HUMAN × COMPUTER INTERACTION

Spring 2017

컴퓨터프로그래밍

프로젝트 : Bind Your Data!

제출 기한 및 참고사항

- 제출 기한 및 딜레이
  - 5월 23일 자정까지
  - 5월 25일 자정까지 딜레이 허용, 하루에 20%씩 감점
- https://cp2017s.snu.ac.kr/project 에 제출 바랍니다.

CSV + Recipe → (Interpreter : Your PROJECT) → SVG

- 주어진 CSV와 Recipe로 SVG를 만드는 인터프리터 만들기
- CSV란?
  - Comma-Separated Values
  - 각 열을 콤마로 구분하는 데이터 기술 형식을 뜻함
  - 이번 프로젝트에서는 명령줄 인자로 n개의 CSV 파일이 들어오게 된다.
    - **이번 프로젝트에서 한번에 주어지는 CSV파일들의 필드의 이름과 그 타입은 동일하다.**
  - 사용할 CSV의 형식은 아래와 같음
    - 이번 프로젝트에서 value의 타입은 string, int, float으로 한정된다.

```
name,age,height // 첫째 줄 = 필드의 이름
string,int,float // 둘째 줄 = 필드의 타입
John,22,173.5 // 셋째 줄 부터 한 줄당 하나의 datum
Mary,23,162.4
Bell,24,155.9
Tom,19,175.4
...
...
```

CSV + Recipe → (Interpreter : Your PROJECT) → SVG

- 주어진 CSV와 Recipe로 SVG를 만드는 인터프리터 만들기
- Recipe란?
  - 주어진 CSV를 SVG로 변환하는 과정이 쓰여진 문서
  - 이번 프로젝트에서는 STDIN으로 입력되어진다
  - Recipe의 print instruction을 통해 해당 명령 전까지 만들어진 SVG를 출력한다

```
append svg
cattr width 500
cattr height 500
selectAll rect
enter 1
dattr width height
cattr height 10
tattr 0 10
print out.html
end
end
end
```

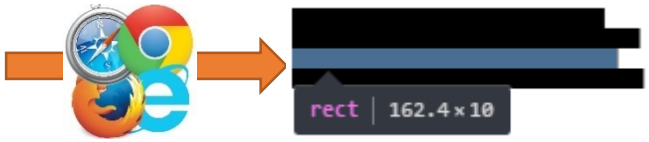
<out.html>

```
<html>
    <svg height="500" width="500">
        <rect height="10" transform="translate(0,0)" width="155.900000"></rect>
        <rect height="10" transform="translate(0,10)" width="173.500000"></rect>
        <rect height="10" transform="translate(0,20)" width="162.400000"></rect>
        <rect height="10" transform="translate(0,30)" width="175.400000"></rect>
    </svg>
</html>
```

CSV + Recipe → (Interpreter : Your PROJECT) → SVG

- 주어진 CSV와 Recipe로 SVG를 만드는 인터프리터 만들기
- SVG란?
  - Scalable Vector Graphics
  - XML(Extensible Markup Language)형식으로 기술된 2차원 벡터 그래픽을 표현하기 위한 파일 형식
  - Chrome등의 웹 브라우저에서 바로 해석하여 렌더링이 가능하다
  - <[태그] {[어트리뷰트 이름]="[어트리뷰트 값]"} x n></[태그]> 형식으로 이루어짐
  - 계층 구조를 가지며 <>와 </> 사이에 자식 노드들이 들어가게 된다.
  - **지금까지의 과제와는 다르게 SVG의 텍스트 값을 비교하는 것이 아닌 렌더링이 제대로 되었는지가 평가 기준이다. (렌더링은 크롬 최신 버전에서 수행)**

```
<html>
    <svg height="500" width="500">
        <rect height="10" transform="translate(0,0)" width="155.900000"></rect>
        <rect height="10" transform="translate(0,10)" width="173.500000"></rect>
        <rect height="10" transform="translate(0,20)" width="162.400000"></rect>
        <rect height="10" transform="translate(0,30)" width="175.400000"></rect>
    </svg>
</html>
```



CSV + Recipe → (Interpreter : Your PROJECT) → SVG

- 주어진 CSV와 Recipe로 SVG를 만드는 인터프리터 만들기
- 인터프리터란?
  - 명령줄 인자로 주어진 CSV들을 숙지하고 STDIN으로 한줄씩 입력되는 Recipe의 Instruction을 해석하여 수행하는 프로그램
  - Instruction을 수행할때마다 인터프리터가 선택하고 있는 SVG 객체가 달라질 수 있음, 인터프리터가 선택하고 있는 SVG 객체들의 목록을 Selection이라고 함
  - Selection에는 세 종류가 있으며 아래와 같다.
    - root-selection : root svg 객체를 선택하고 있는 경우
    - single-selection : root svg 객체가 아닌 다른 svg 객체를 선택하고 있는 경우
    - multiple-selection : 같은 부모를 둔 svg 객체들을 선택하고 있는 경우
      - ※ multiple-selection이 0개 또는 1개의 svg 객체를 선택하고 있을 수도 있다.
  - "html"을 태그 이름으로 하는 root svg 객체는 프로그램 수행시 자동으로 생성되어야 하며, 인터프리터의 초기 selection은 무조건 이 객체를 선택하는 root-selection이다.

Instructions : append

- append <svg-tag-name>
  - (root-selection || single-selection) → (single-selection)
- Selection의 child로 <svg-tag-name> tag를 가진 새로운 svg 객체를 자식 계층의 마지막에 추가한다.
- Instruction 수행 후 selection은 새로 추가된 svg를 선택한다.

```
<- selection scope
<_:selection>
</_:selection>

// append <svg-tag-name>

<_>
    <- selection scope
    <svg-tag-name:selection>
    </svg-tag-name:selection>
</_>
```

Instructions : select

- select <svg-tag-name>
  - (root-selection || single-selection) → (single-selection)
- Selection의 child중 <svg-tag-name>을 가지는 객체를 선택한다.
  - ※ <svg-tag-name>을 가지는 child가 단 하나 있음이 보장되어있다.
- Instruction 수행 후 selection은 해당 child를 선택한다.

```
<- selection scope
<_:selection>
    <svg-tag-name>
    </svg-tag-name>
</_:selection>

// select <svg-tag-name>

<_>
    <- selection scope
    <svg-tag-name:selection>
    </svg-tag-name:selection>
</_>
```

Instructions : selectAll

- selectAll <svg-tag-name>
  - (root-selection || single-selection) → (multiple-selection)
- Selection의 child중 <svg-tag-name>을 가지는 객체를 모두 선택한다.
  - ※ <svg-tag-name>을 가지는 child가 없어도, 1개만 있어도 무방하다
- Instruction 수행 후 selection은 해당 child들을 선택한다.

Instructions : selectAll

```
<- selection scope
<_:selection>
    <svg-tag-name>
    </svg-tag-name>
    <not-svg-tag-name>
    </not-svg-tag-name>
    <svg-tag-name>
    </svg-tag-name>
</_:selection>

// selectAll <svg-tag-name>

<_>
    <- selection scope
    <svg-tag-name:selection>
    </svg-tag-name:selection>
    <not-svg-tag-name>
    </not-svg-tag-name>
    <svg-tag-name:selection>
    </svg-tag-name:selection>
</_>
```

```
<- selection scope
<_:selection>
    <not-svg-tag-name>
    </not-svg-tag-name>
    <not-svg-tag-name>
    </not-svg-tag-name>
    <not-svg-tag-name>
    </not-svg-tag-name>
</_:selection>

// selectAll <svg-tag-name>

<_>
    <- selection scope
    <not-svg-tag-name>
    </not-svg-tag-name>
    <not-svg-tag-name>
    </not-svg-tag-name>
    <not-svg-tag-name>
    </not-svg-tag-name>
</_>
```

Instructions : remove

- remove
  - (single-selection || multiple-selection) → (single-selection || root-selection)
- 현재 selection이 선택하고 있는 모든 svg 객체를 삭제한다
- Instruction 수행 후 selection은 삭제된 svg 객체(들)의 부모를 선택한다.

```
<_>
    <- selection scope
    <__:selection>
    </__:selection>
    <___>
    </___>
    <__:selection>
    </__:selection>
</_>

// remove

<- selection scope
<_:selection>
    <___>
    </___>
</_:selection>
```

Instructions : end

- end
  - (single-selection || multiple-selection || root-selection) → (single-selection || root-selection)
- Selection이 현재 selection의 부모 svg 객체를 선택하도록 한다.
- 만약 현재 selection이 root-selection이라면 프로그램을 종료한다.

```
<_>
    <- selection scope
    <__:selection>
    </__:selection>
</_>

// end

<- selection scope
<_:selection>
    <__>
    </__>
</_:selection>
```

Instructions : enter

- enter <csv-index>
  - (multiple-selection) → (multiple-selection)
- 현재 selection의 svg 객체들이 바인딩하고 있는 데이터들과 <csv-index>번째의 csv 파일의 데이터들을 비교한다.
- csv 파일의 데이터 중 그 unique-id가 svg 객체에 바인딩 된 데이터의 unique-id와 같지 않은 데이터들에 바인딩 된 svg 객체들을 새로 만들어 같은 계층에 마지막에 추가한다.
  - **이 때, unique-id를 오름차순 정렬하여 바인딩 해야한다.**
- unique-id는 첫번째 필드의 value다.
  - 다시 말해 csv 파일의 첫번째 필드가 primary key가 된다.
  - 첫번째 필드의 모든 value는 unique함(중복값이 존재하지 않음)이 보장된다.
- Instruction 후 selection은 새로 추가된 svg 객체들이다.

Instructions : enter (figure, rect의 닫는 태그 생략)

- enter <csv-index>
  - (multiple-selection) → (multiple-selection)
  - selection**이 0개일 때에도** enter**가 가능하다는 것에 주의**

```
./interpreter people1.csv people2.csv

<people1.csv>
name,age,height
string,int,float
John,22,173.5
Mary,23,162.4
Bell,24,155.9
Tom,19,175.4

<people2.csv>
name,age,height
string,int,float
Mary,23,162.4
Bell,24,155.9
Tom,19,168.2
Bob,27,172.4
```

```
<html>
    <- selection scope
    <svg:selection>
    </svg:selection>
</html>

selectAll rect
enter 1

<html>
    <svg>
        <- selection scope
        <rect:selection> <- binded with Bell in people1.csv
        <rect:selection> <- binded with John in people1.csv
        <rect:selection> <- binded with Mary in people1.csv
        <rect:selection> <- binded with Tom in people1.csv
    </svg>
</html>
```

Instructions : update

- update <csv-index>
  - (multiple-selection) → (multiple-selection)
- 현재 selection의 svg 객체들이 바인딩하고 있는 데이터들과 <csv-index>번째의 csv 파일의 데이터들을 비교한다.
- csv 파일의 데이터 중 그 unique-id가 svg 객체에 바인딩된 데이터의 unique-id와 같은 데이터들을 해당 svg 객체에 다시 바인딩한다.
- Instruction 후 selection은 다시 바인딩된 svg 객체들이다.

Instructions : update (figure, rect의 닫는 태그 생략)

- update <csv-index>
  - (multiple-selection) → (multiple-selection)

```
./interpreter people1.csv people2.csv

<people1.csv>
name,age,height
string,int,float
John,22,173.5
Mary,23,162.4
Bell,24,155.9
Tom,19,175.4

<people2.csv>
name,age,height
string,int,float
Mary,23,162.4
Bell,24,155.9
Tom,19,175.4
Bob,27,172.4
```

```
<html>
    <svg>
        <- selection scope
        <rect:selection> <- binded with Bell in people1.csv
        <rect:selection> <- binded with John in people1.csv
        <rect:selection> <- binded with Mary in people1.csv
        <rect:selection> <- binded with Tom in people1.csv
    </svg>
</html>

update 2

<html>
    <svg>
        <- selection scope
        <rect:selection> <- binded with Bell in people2.csv
        <rect> <- binded with John in people1.csv
        <rect:selection> <- binded with Mary in people2.csv
        <rect:selection> <- binded with Tom in people2.csv
    </svg>
</html>
```

Instructions : exit

- exit <csv-index>
  - (multiple-selection) → (multiple-selection)
- 현재 selection의 svg 객체들이 바인딩하고 있는 데이터들과 <csv-index>번째의 csv 파일의 데이터들을 비교한다.
- selection은 svg 객체에 바인딩 된 데이터 중 그 unique-id가 csv 파일의 데이터의 unique-id와 같지 않은 svg 객체들을 선택한다.

Instructions : exit (figure, rect의 닫는 태그 생략)

- exit <csv-index>
  - (multiple-selection) → (multiple-selection)

```
./interpreter people1.csv people2.csv

<people1.csv>
name,age,height
string,int,float
John,22,173.5
Mary,23,162.4
Bell,24,155.9
Tom,19,175.4

<people2.csv>
name,age,height
string,int,float
Mary,23,162.4
Bell,24,155.9
Tom,19,175.4
Bob,27,172.4
```

```
<html>
    <svg>
        <- selection scope
        <rect:selection> <- binded with Bell in people1.csv
        <rect:selection> <- binded with John in people1.csv
        <rect:selection> <- binded with Mary in people1.csv
        <rect:selection> <- binded with Tom in people1.csv
    </svg>
</html>

exit 2

<html>
    <svg>
        <- selection scope
        <rect> <- binded with Bell in people1.csv
        <rect:selection> <- binded with John in people1.csv
        <rect> <- binded with Mary in people1.csv
        <rect> <- binded with Tom in people1.csv
    </svg>
</html>
```

Instructions : cattr

- cattr <svg-attr-name> <svg-attr-value>
- Selection이 선택하고 있는 svg 객체(들)의 attribute를 수정한다.
  - attribute가 없다면 새로 추가하고, 있다면 값을 수정한다.

```
<_>
    <- selection scope
    <__:selection>
    </__:selection>
    <___>
    </___>
</_>

// cattr <svg-attr-name> <svg-attr-value>

<_:selection>
    <- selection scope
    <__:selection <svg-attr-name>="<svg-attr-value>">
    </__:selection  <svg-attr-name>="<svg-attr-value>">
    <___>
    </___>
</_:selection>
```

Instructions : tattr

- tattr <x-multiplier> <y-multiplier>
- Selection이 선택하고 있는 svg 객체(들)을 translate한다.
- svg 객체의 translate는 해당 객체에 transform="translate(x, y)" 형식의 attribute를 넣음으로써 수행할 수 있다.
  - 이 때 x와 y에는 <_-multiplier> * index에 해당하는 수치가 들어간다.
  - index는 해당 svg객체가 selection에서 몇 번째에 위치하는지에 대한 순서 정보
  - transform attribute가 없다면 새로 추가하고, 있다면 값을 수정한다.

Instructions : tattr (figure, rect의 닫는 태그 생략)

```
<html>
    <svg>
        <- selection scope
        <rect:selection> <- binded with Bell in people2.csv
        <rect> <- binded with John in people1.csv
        <rect:selection> <- binded with Mary in people2.csv
        <rect:selection> <- binded with Tom in people2.csv
    </svg>
</html>

tattr 10 20


<html>
    <svg>
        <- selection scope
        <rect:selection transform="translate(0,0)"> <- binded with Bell in people2.csv
        <rect> <- binded with John in people1.csv
        <rect:selection transform="translate(10,20)"> <- binded with Mary in people2.csv
        <rect:selection transform="translate(20,40)"> <- binded with Tom in people2.csv
    </svg>
</html>
```

Instructions : dattr

- dattr <svg-attr-name> <datum-field-name> [<mul>] [<add>]
- Selection이 선택하고 있는 svg 객체(들)의 attribute를 그 객체가 바인딩하고 있는 데이터의 필드를 참조하여 수정한다.
  - attribute가 없다면 새로 추가하고, 있다면 값을 수정한다.
- 만약 <datum-field-name>에 해당하는 field의 타입이 int거나 float이고, [<mul>]과 [<add>]가 주어졌다면, value * <mul> + <add>의 값으로 수정한다.
  - dattr은 [<mul>]과 [<add>] 둘다 주어지지 않거나, [<mul>]만 주어지거나, [<mul>]과 [<add>]가 동시에 주어지는 세 가지의 입력이 가능함.
  - <datum-field-name>의 타입이 string인데 [<mul>]과 [<add>]를 주는 dattr 명령은 들어오지 않음이 보장된다.
- Selection이 선택하고 있는 svg 객체(들) 중 하나라도 데이터가 바인딩 되어있지 않은 경우에 dattr 명령은 들어오지 않음이 보장된다.

Instructions : dattr (figure, rect의 닫는 태그 생략)

```
<html>
    <svg>
        <- selection scope
        <rect:selection> <- binded with Bell in people2.csv
        <rect> <- binded with John in people1.csv
        <rect:selection> <- binded with Mary in people2.csv
        <rect:selection> <- binded with Tom in people2.csv
    </svg>
</html>

dattr width height 2 10

<html>
    <svg>
        <- selection scope
        <rect:selection width="321.8"> <- binded with Bell in people2.csv
        <rect> <- binded with John in people1.csv
        <rect:selection width="334.8"> <- binded with Mary in people2.csv
        <rect:selection width="346.4"> <- binded with Tom in people2.csv
    </svg>
</html>
```

Instructions : print

- print <output-name>
- root svg 객체들을 <output-name> 파일에 출력한다.
- 출력 양식은 아래의 양식을 재귀적으로 호출하면 된다.
  - <tag attr1="attr1-value" attr2="attr2-value" ...>
    (tag의 자식들 출력)
    </tag>

예시 입력과 출력 (1)

```
./interpreter 1.csv 2.csv

<1.csv>

name,value
string,int
tomato,2000
apple,1500
orange,3000
grape,4000

<2.csv>

name,value
string,int
tomato,1400
orange,3200
grape,3800
kiwi,1500
```

```
append svg
cattr width 1000
cattr height 500
print 1.html
selectAll rect
enter 1
dattr name name
dattr width value 0.1 10
cattr height 10
tattr 0 13
print 2.html
end
selectAll rect
update 2
dattr width value 0.2 10
cattr fill steelblue
print 3.html
end
selectAll rect
exit 2
dattr width value 0.05 10
print 4.html
remove
end
print 5.html
end
```

예시 입력과 출력 (1)

<1.html>

```
<html><svg height="500" width="1000"></svg></html>
```

<2.html>

```
<html><svg height="500" width="1000"><rect height="10" name="apple" transform="translate(0,0)" width="160.000000"></rect><rect height="10" name="grape" transform="translate(0,13)" width="410.000000"></rect><rect height="10" name="orange" transform="translate(0,26)" width="310.000000"></rect><rect height="10" name="tomato" transform="translate(0,39)" width="210.000000"></rect></svg></html>
```

예시 입력과 출력 (1)

<3.html>

```
<html><svg height="500" width="1000"><rect height="10" name="apple" transform="translate(0,0)" width="160.00000"></rect><rect fill="steelblue" height="10" name="grape" transform="translate(0,13)" width="770.000000"></rect><rect fill="steelblue" height="10" name="orange" transform="translate(0,26)" width="650.000000"></rect><rect fill="steelblue" height="10" name="tomato" transform="translate(0,39)" width="290.000000"></rect></svg></html>
```

<4.html>

```
<html><svg height="500" width="1000"><rect height="10" name="apple" transform="translate(0,0)" width="85.00000"></rect><rect fill="steelblue" height="10" name="grape" transform="translate(0,13)" width="770.000000"></rect><rect fill="steelblue" height="10" name="orange" transform="translate(0,26)" width="650.000000"></rect><rect fill="steelblue" height="10" name="tomato" transform="translate(0,39)" width="290.000000"></rect></svg></html>
```

예시 입력과 출력 (1)

<5.html>

```
<html><svg height="500" width="1000"><rect fill="steelblue" height="10" name="grape" transform="translate(0,13)" width="770.000000"></rect><rect fill="steelblue" height="10" name="orange" transform="translate(0,26)" width="650.000000"></rect><rect fill="steelblue" height="10" name="tomato" transform="translate(0,39)" width="290.000000"></rect></svg></html>
```

예시 입력과 출력 (2)

```
./interpreter 1.csv 2.csv

<1.csv>

name,length,width
string,float,float
a,5.1,3.5
b,4.9,3.0
c,4.7,3.2
d,4.6,3.1
e,5.0,3.6
f,5.4,3.9
g,4.6,3.4
h,5.0,3.4
i,4.4,2.9
j,4.9,3.1
k,5.4,3.7
l,4.8,3.4
m,4.8,3.0
n,4.3,3.0
o,5.8,4.0
p,5.7,4.4
q,5.4,3.9
r,5.1,3.5
s,5.7,3.8
t,5.1,3.8
```

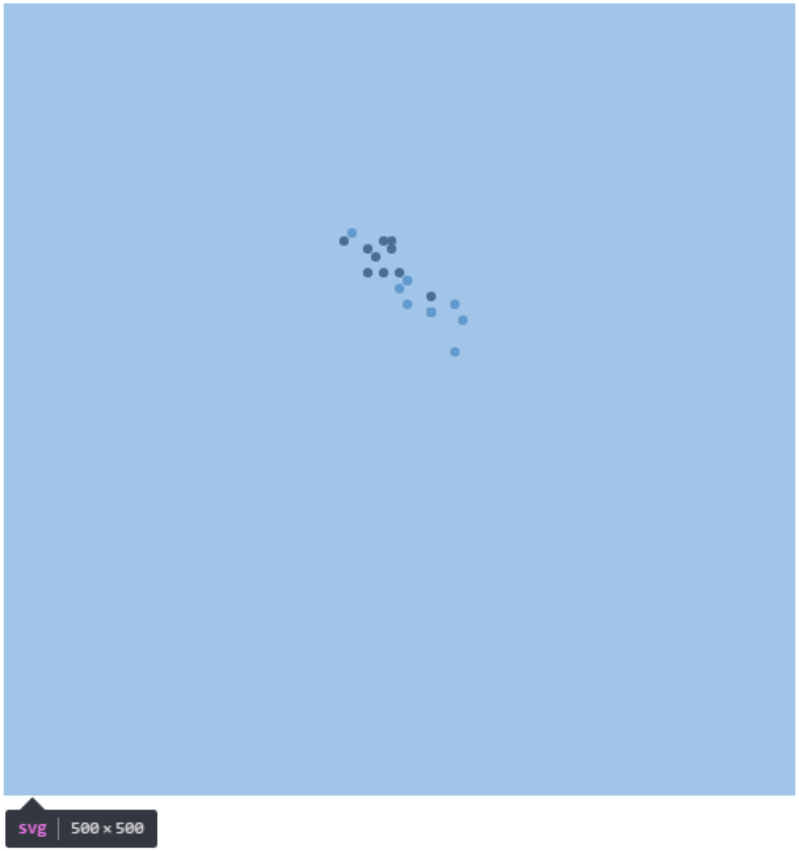
```
<2.csv>

name,length,width
string,float,float
q,6.0,2.2
w,6.9,3.2
e,5.6,2.8
r,7.7,2.8
t,6.3,2.7
y,6.7,3.3
u,7.2,3.2
i,6.2,2.8
o,6.1,3.0
p,6.4,2.8
a,7.2,3.0
s,7.4,2.8
```

```
append svg
cattr width 500
cattr height 500
selectAll circle
enter 1
cattr r 3
dattr name name
dattr cx length 50
dattr cy width 50
print 1.html
end
selectAll circle
update 2
cattr fill steelblue
dattr name name
print 2.html
dattr cx length 50
dattr cy width 50
print 3.html
end
selectAll circle
exit 2
remove
end
print 4.html
end
```

예시 입력과 출력 (2)

<1.html>

<2.html>



예시 입력과 출력 (2)

<3.html>

<4.html>