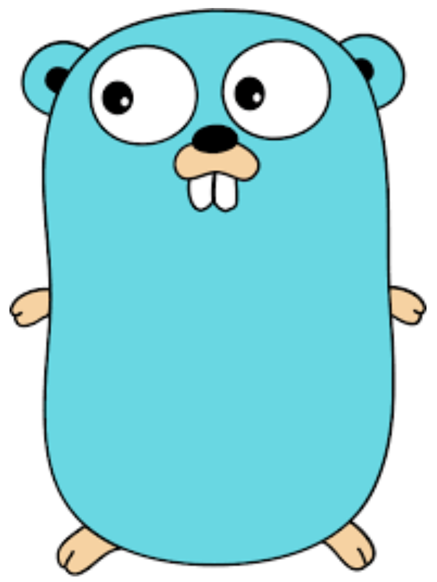
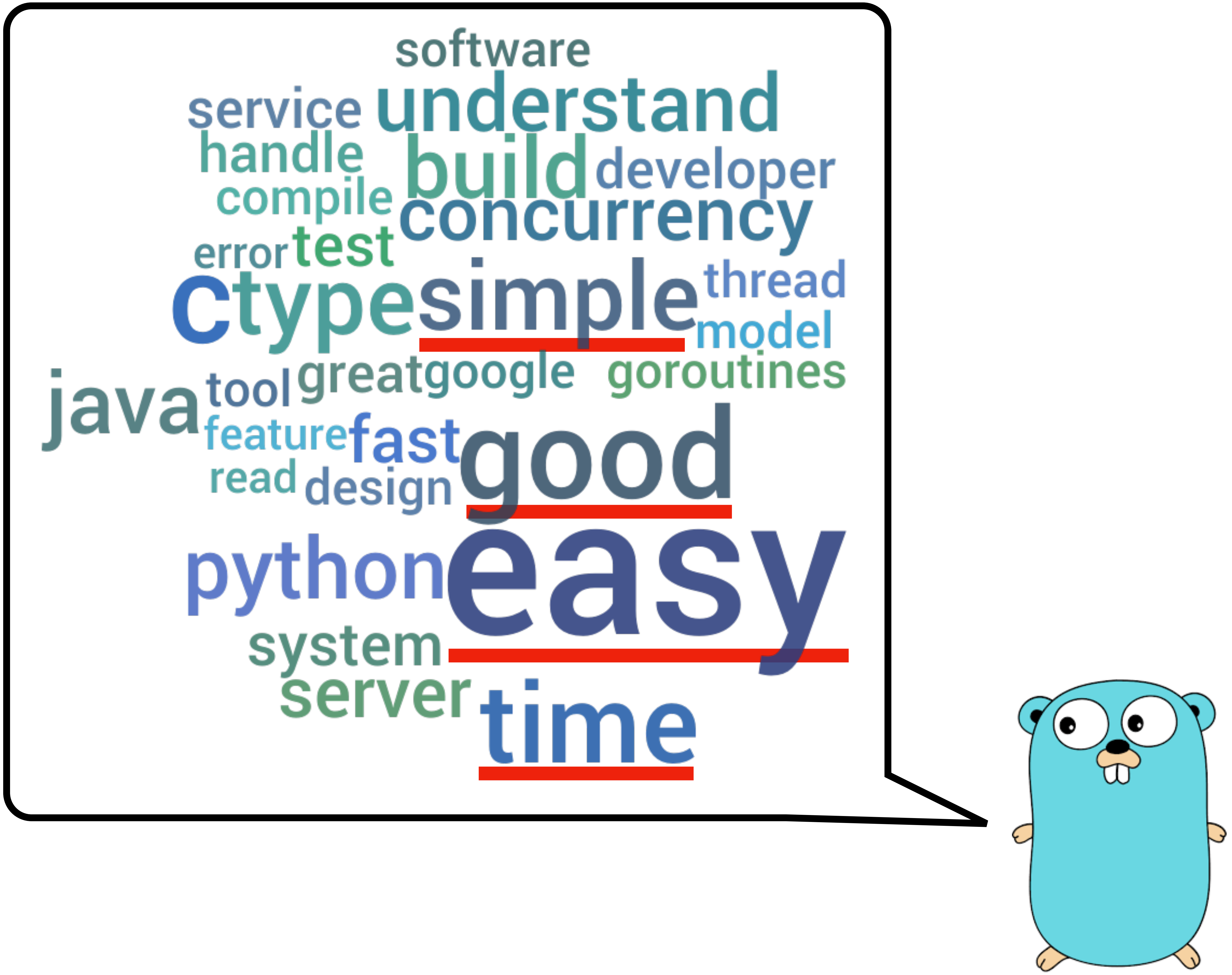
Project-Driven Journey to Learning Go

GitHub/elissalim

SPdigital

Why Go?




software
service
understand
handle
build
developer
compile
concurrency
error
test
c
type
simple
thread
model
java
tool
great
google
goroutines
feature
fast
good
read
design
python
easy
system
server
time

TextMining

Text Mining

1. Scrape individual online review
2. Combine reviews

TextProcessing

Text Processing

3. Remove non-alphanumeric characters
4. Apply lowercase

Natural Language Processing

5. Apply lemmatization
6. Remove stop words

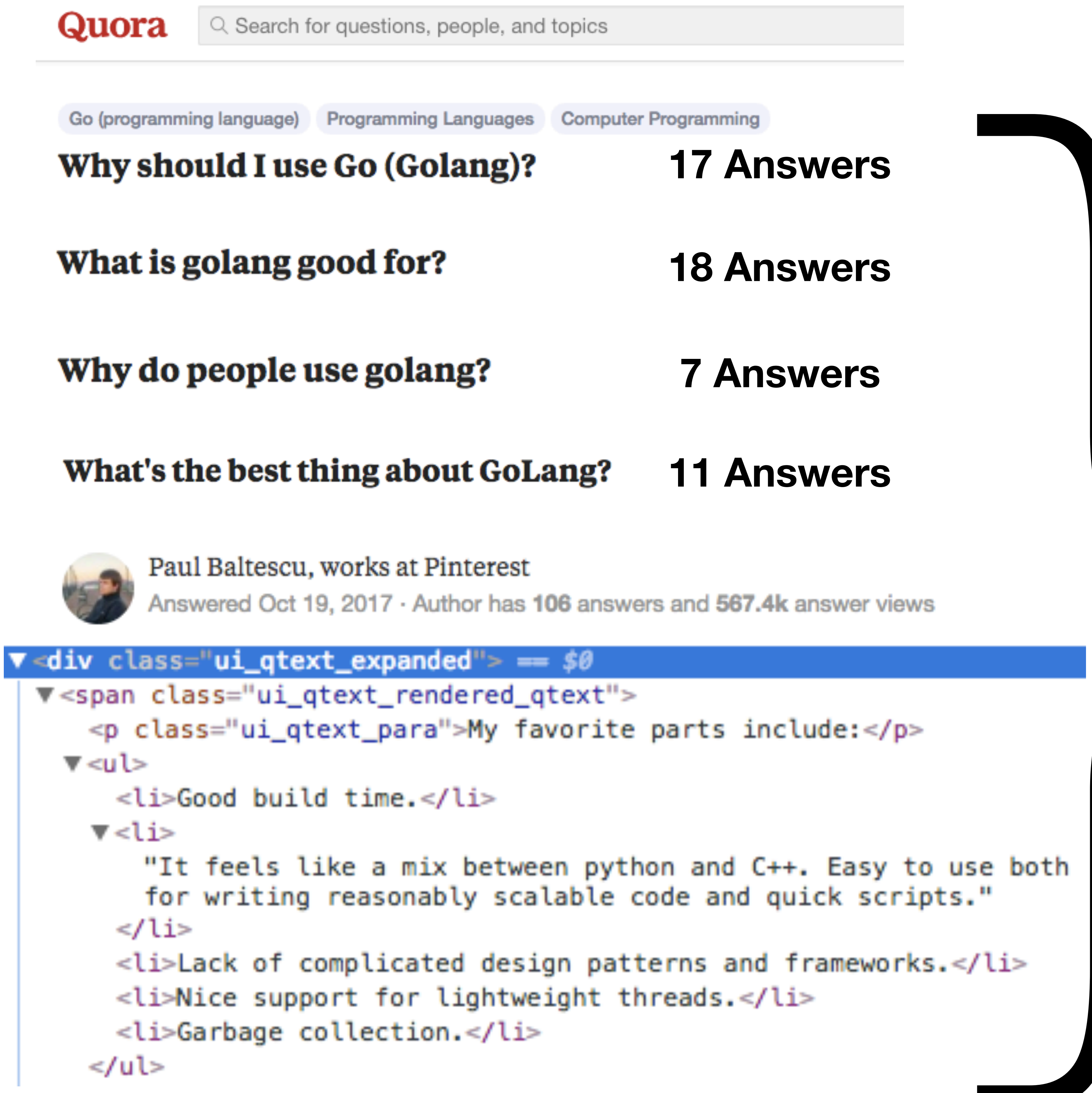
WordCloud

Visualization

7. Create word cloud

Text Mining

1. **Scrape individual online review**



Quora Search for questions, people, and topics

Go (programming language) · Programming Languages · Computer Programming

Why should I use Go (Golang)? 17 Answers

What is golang good for? 18 Answers

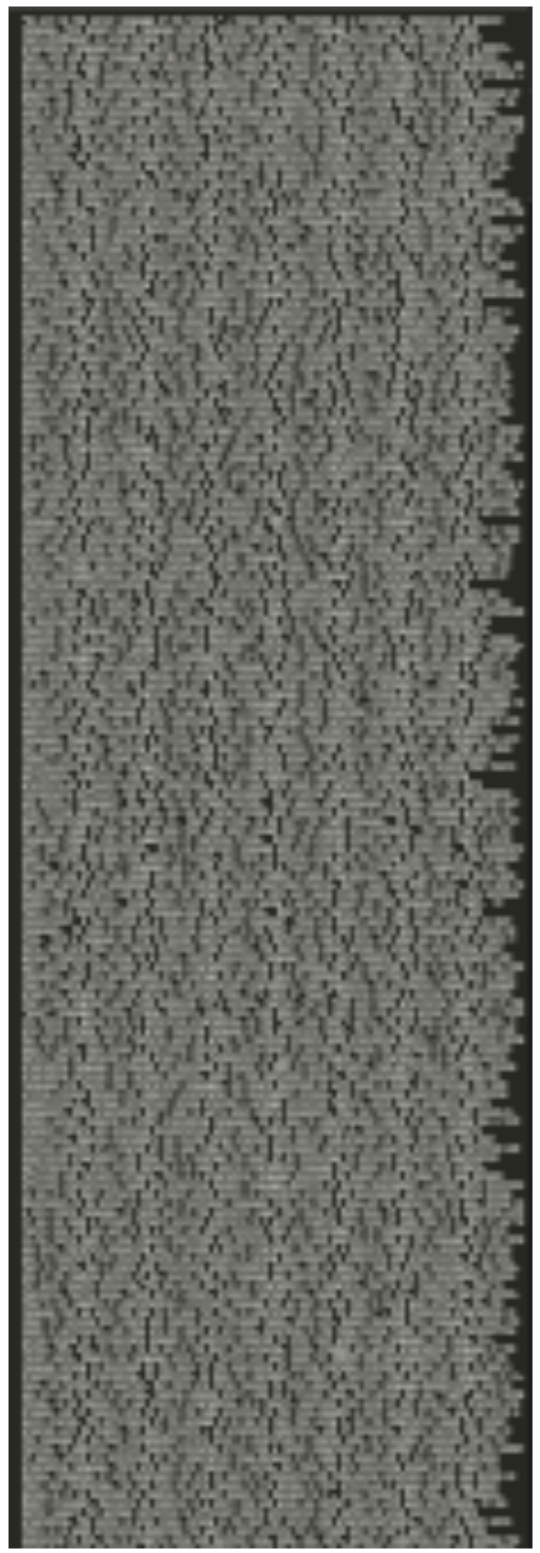
Why do people use golang? 7 Answers

What's the best thing about GoLang? 11 Answers

Paul Baltescu, works at Pinterest
Answered Oct 19, 2017 · Author has **106** answers and **567.4k** answer views

```
▼<div class="ui_qtext_expanded"> == $0
  ▼<span class="ui_qtext_rendered_qtext">
     <p class="ui_qtext_para">My favorite parts include:</p>
    ▼<ul>
       <li>Good build time.</li>
      ▼<li>
         "It feels like a mix between python and C++. Easy to use both
         for writing reasonably scalable code and quick scripts."
       </li>
       <li>Lack of complicated design patterns and frameworks.</li>
       <li>Nice support for lightweight threads.</li>
       <li>Garbage collection.</li>
     </ul>
```

2. **Combine reviews**



Text Mining

External library: github.com/PuerkitoBio/goquery

```
func TextMining() string {
	result := ""
	websites := websitesList()
	for _, v := range websites {
		doc, err := goquery.NewDocument(v)
		if err != nil {
			log.Println(err)
		}
		text := doc.Find(".ui_qtext_expanded").Text()
		result = result + " " + text
	}
	return result
}
```

Text Processing

3. **Remove non-alphanumeric characters**

```
content := textmining.TextMining()

reg, err := regexp.Compile("[^a-zA-Z0-9]+")
if err != nil {
	log.Println(err)
}
processedString := reg.ReplaceAllString(content, " ")
```

4. **Apply lowercase**

```
lowerString := strings.ToLower(processedString)
```

“My favorite parts include: Good build time…”

my favorite parts include good build time

Natural Language Processing

5. Apply lemmatization

External library: **github.com/aaaton/golem**

```go
func lemmatizeWords(reviews []string) []string {
	lemmatizer, err := golem.New("english")
	if err != nil {
		log.Println(err)
	}
	reviewsLength := len(reviews)
	var result []string
	for i := 0; i < reviewsLength; i++ {
		word, err := lemmatizer.Lemma(reviews[i])
		if err != nil {
			result = append(result, reviews[i])
		}
		result = append(result, word)
	}
	return result
}
```

“Compared to other programming languages, Go is easier to understand.”

compare to other program language go be easy to understand

Natural Language Processing

6. Remove stop words

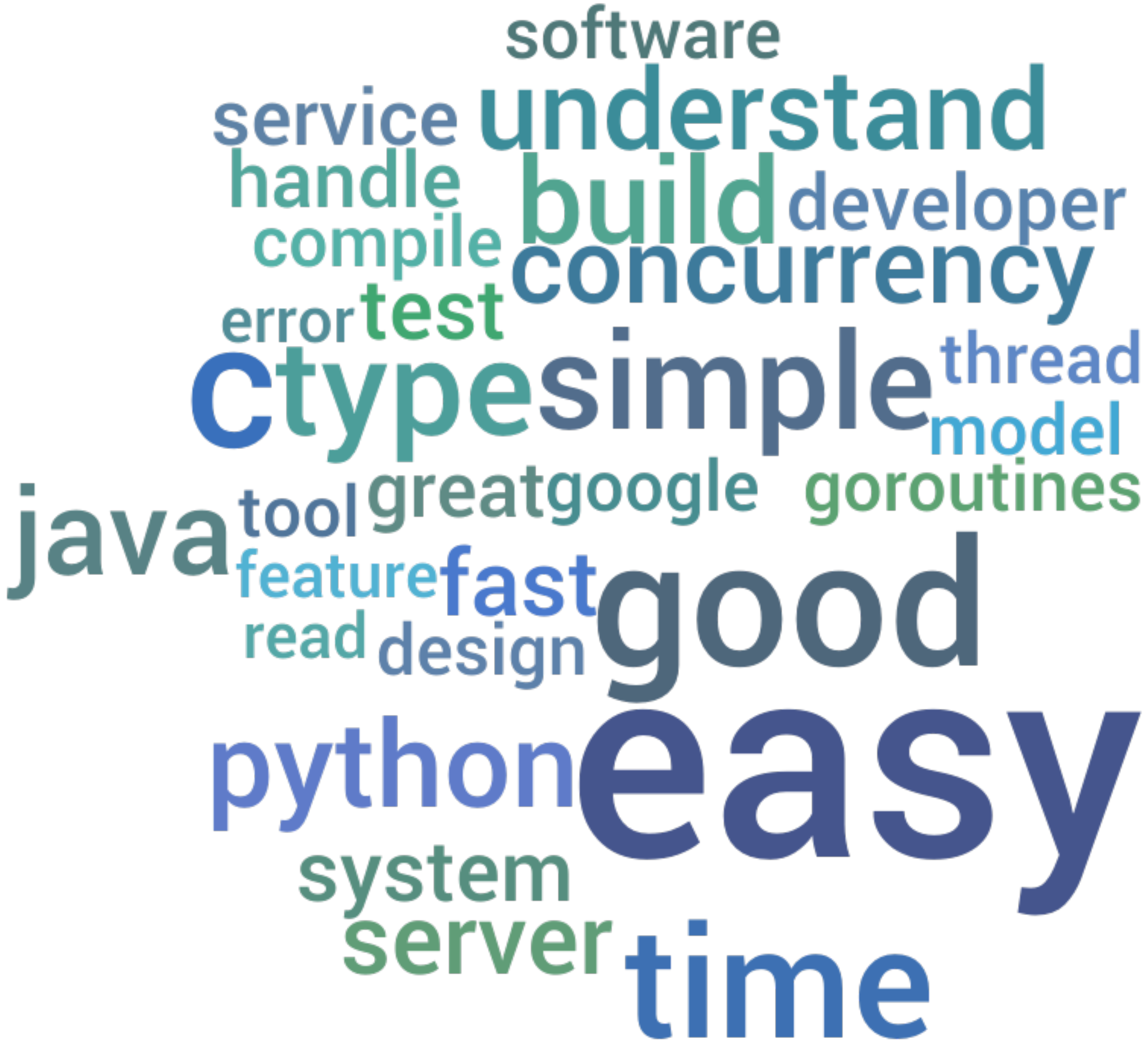
```
stopWords := stopWordsList()
removeStopWords := removeStopWords(stopWords, reviewsListCount)

func removeStopWords(stopWords []string, reviews map[string]int) map[string]int {
	for _, v := range stopWords {
		if reviews[v] > 0 {
		   delete(reviews, v)
		}
	}
	return reviews
}
```

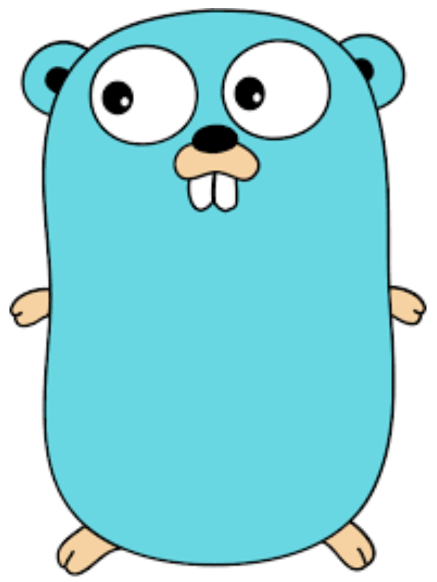
"Compared to other programming languages, Go is easier to understand."

compare:1 easy:1 understand:1

Visualization

7. **Create word cloud**





Visualization

7. Create word cloud

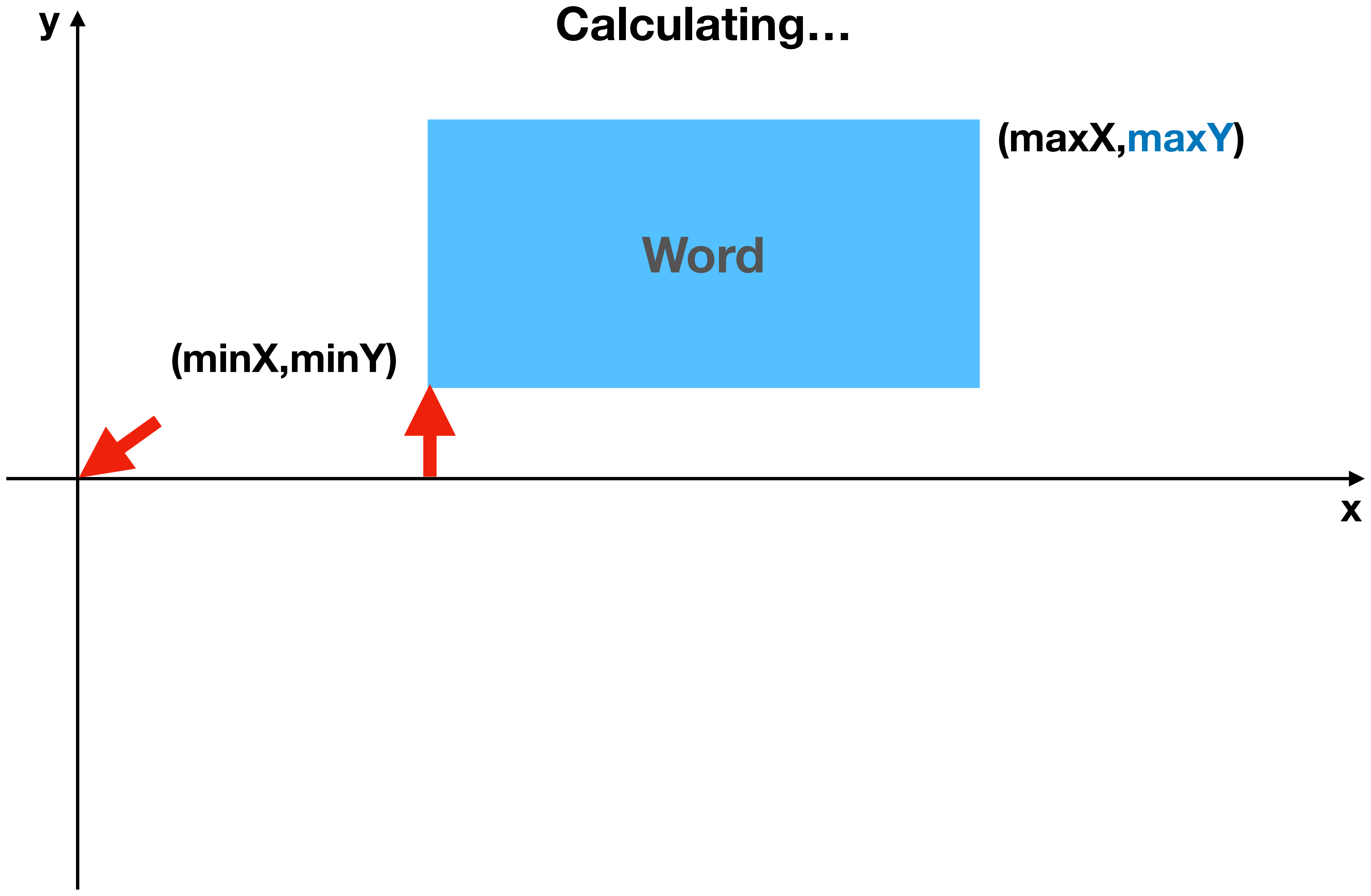
Determine:

1. Image size
2. Number of words to draw
3. Order of words to draw
4. Font
5. Font size
6. Where to draw each word
    - Word SHOULD NOT be outside of image
    - Word SHOULD NOT overlap other drawn words

Visualization

7. Create word cloud

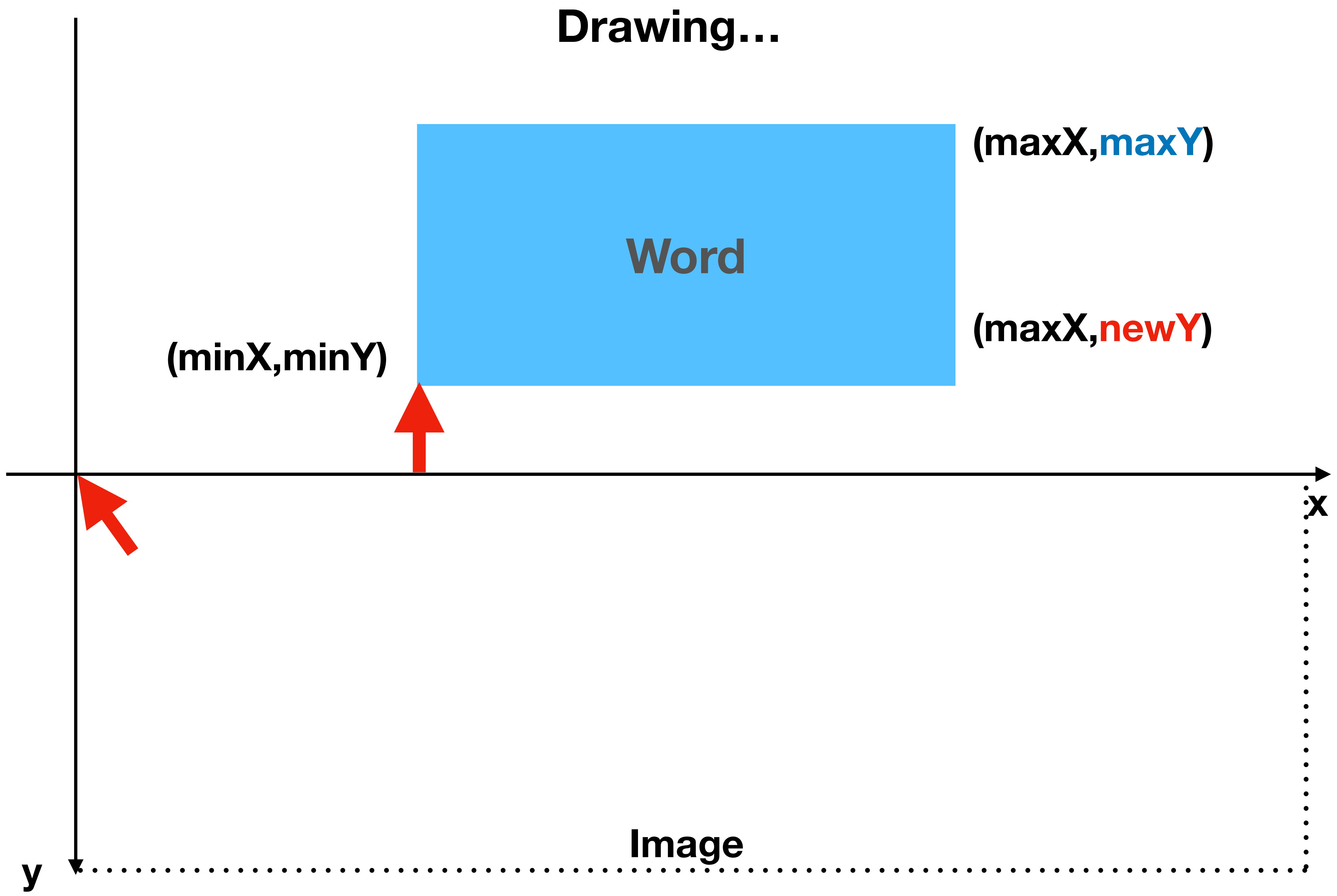
- Word SHOULD NOT be outside of image



Visualization

7. Create word cloud

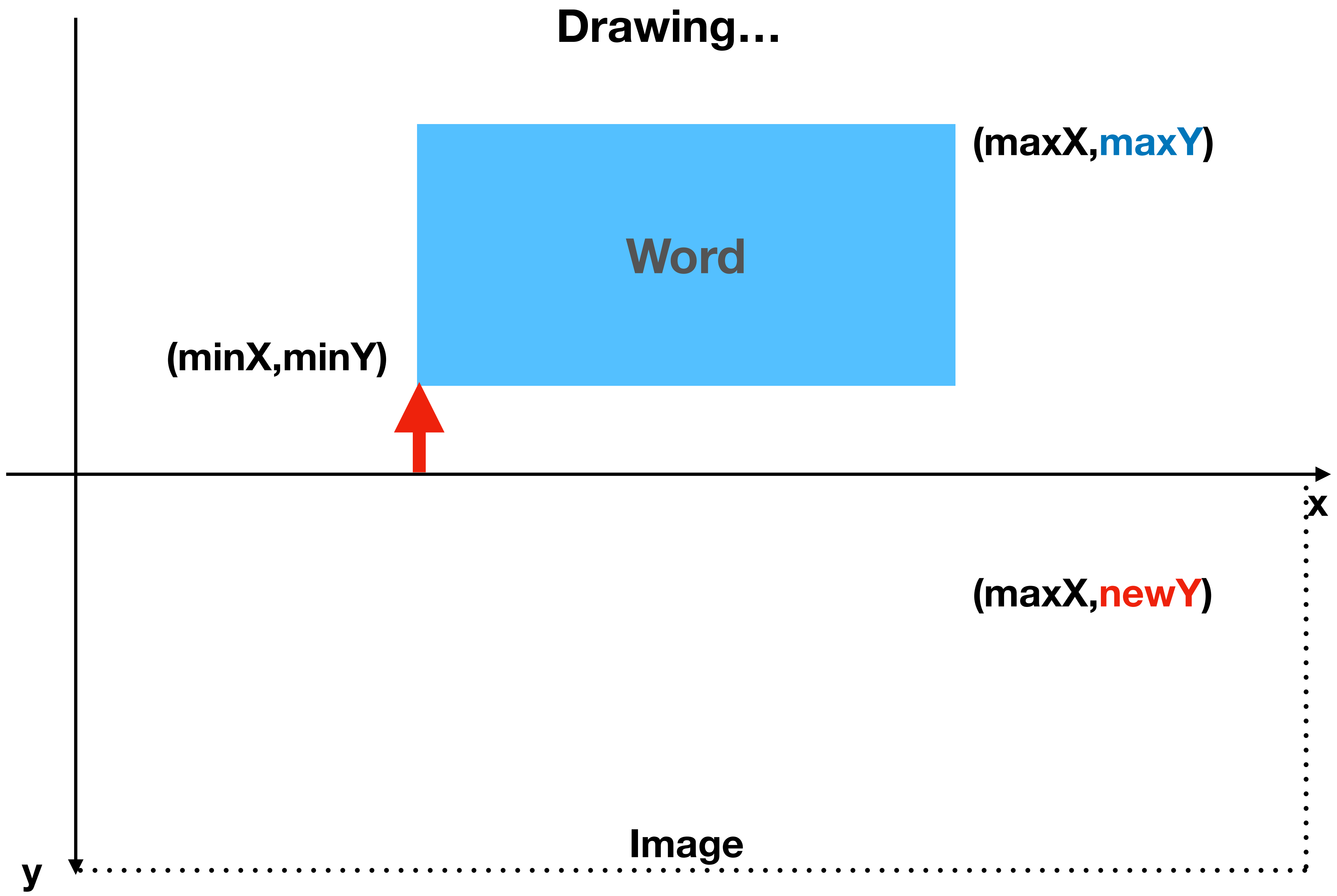
- Word SHOULD NOT be outside of image



Visualization

7. Create word cloud

- Word SHOULD NOT be outside of image



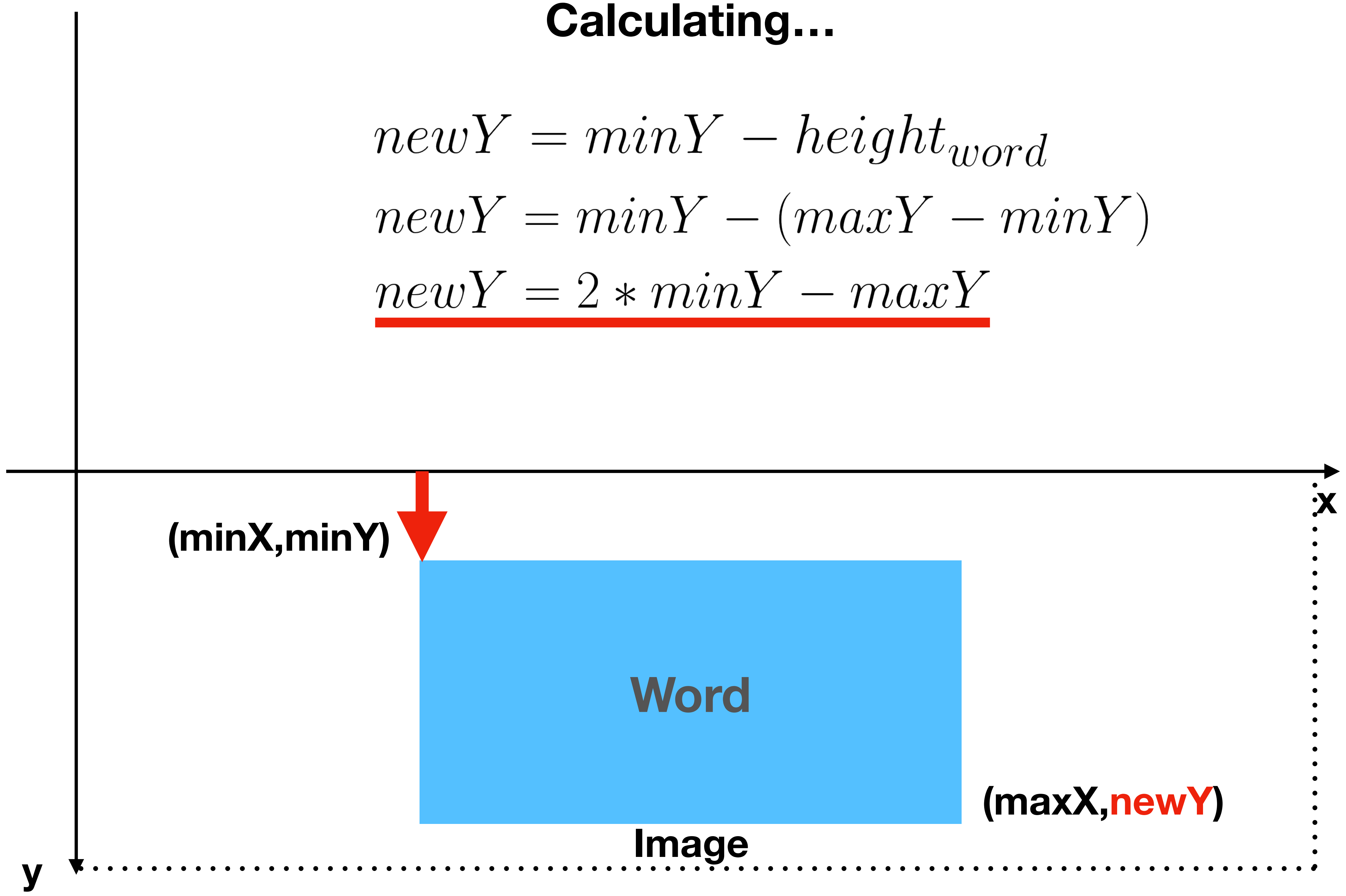
Visualization

7. Create word cloud

- Word SHOULD NOT be outside of image

Calculating…

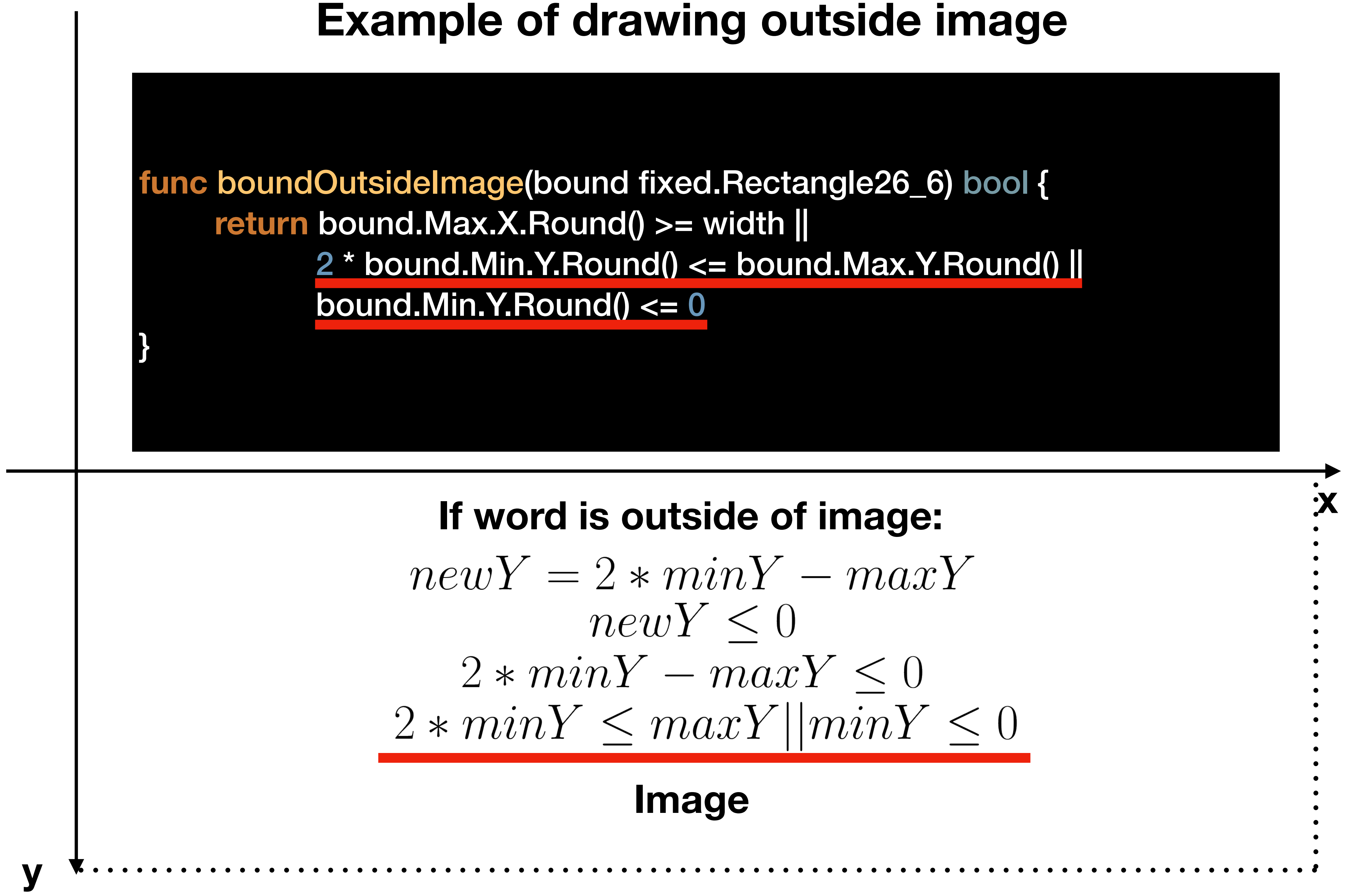
$$newY = minY - height_{word}$$
$$newY = minY - (maxY - minY)$$
$$\underline{newY = 2 * minY - maxY}$$



Visualization

7. Create word cloud
   - Word SHOULD NOT be outside of image

Example of drawing outside image



```
func boundOutsideImage(bound fixed.Rectangle26_6) bool {
        return bound.Max.X.Round() >= width ||
                2 * bound.Min.Y.Round() <= bound.Max.Y.Round() ||
                bound.Min.Y.Round() <= 0
}
```

If word is outside of image:

$$newY = 2 * minY - maxY$$

$$newY \leq 0$$

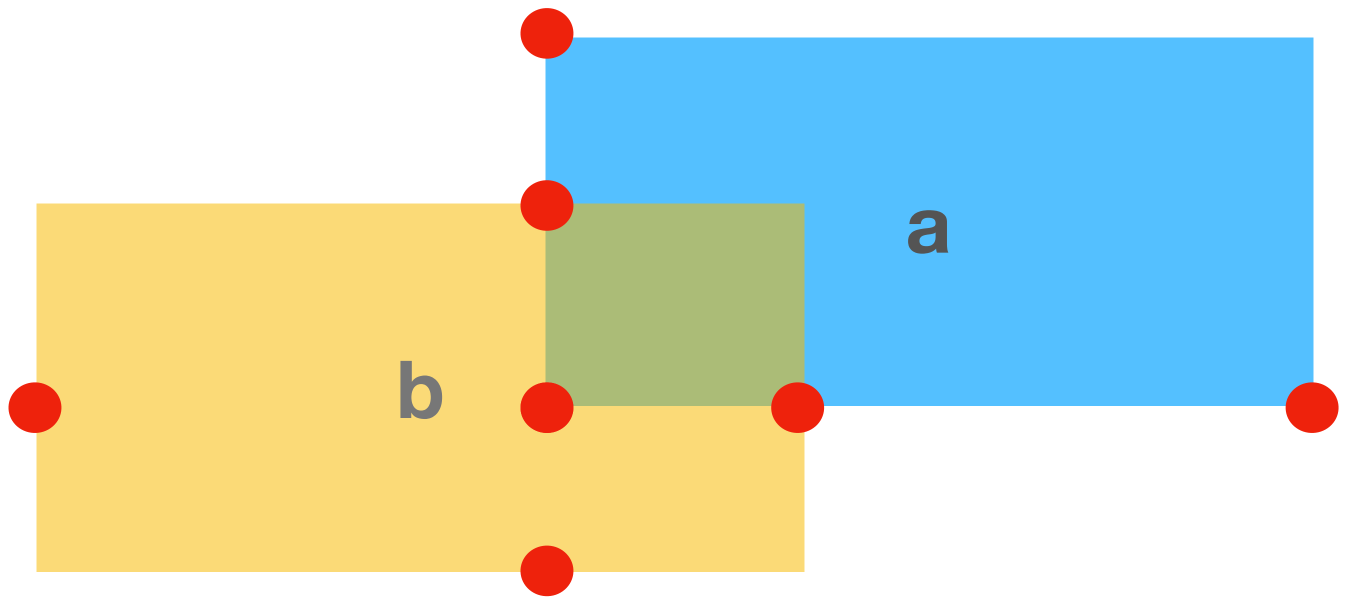
$$2 * minY - maxY \leq 0$$

$$\underline{2 * minY \leq maxY || minY \leq 0}$$

Visualization

7. Create word cloud
   - Word SHOULD NOT overlap other drawn words

Example of colliding

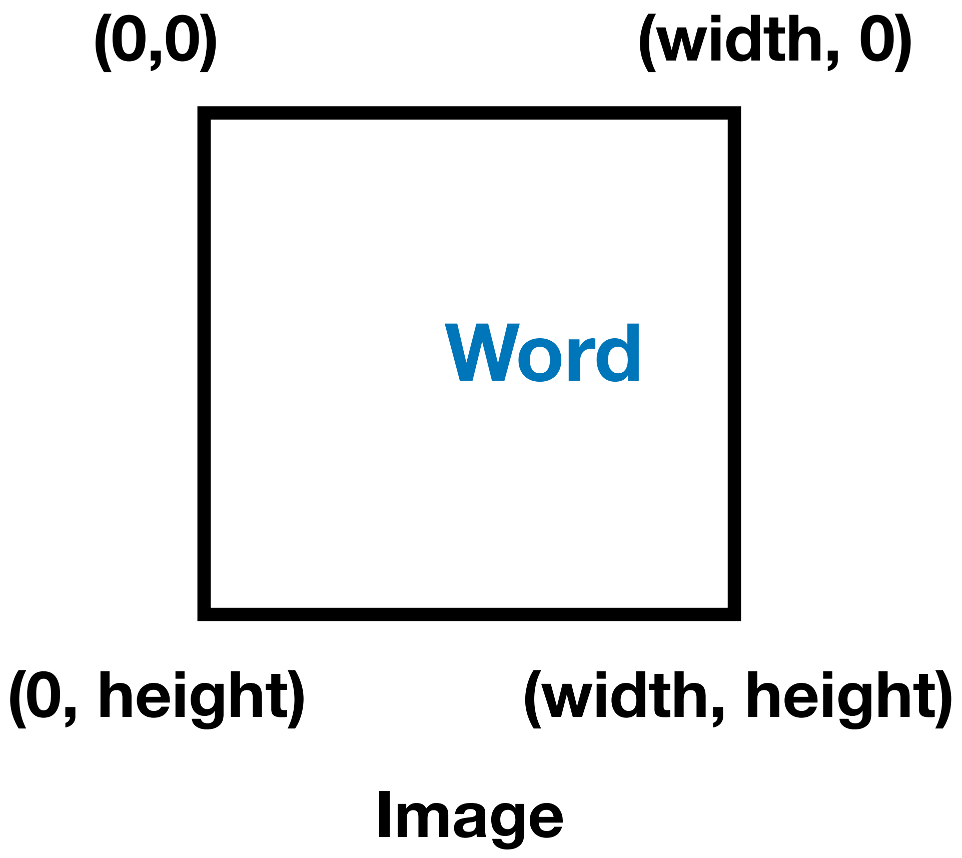


```
func colliding(a, b fixed.Rectangle26_6) bool {
	return a.Min.X <= b.Max.X && a.Max.X >= b.Min.X &&
a.Min.Y <= b.Max.Y && a.Max.Y >= b.Min.Y
}
```

Visualization

7. Create word cloud

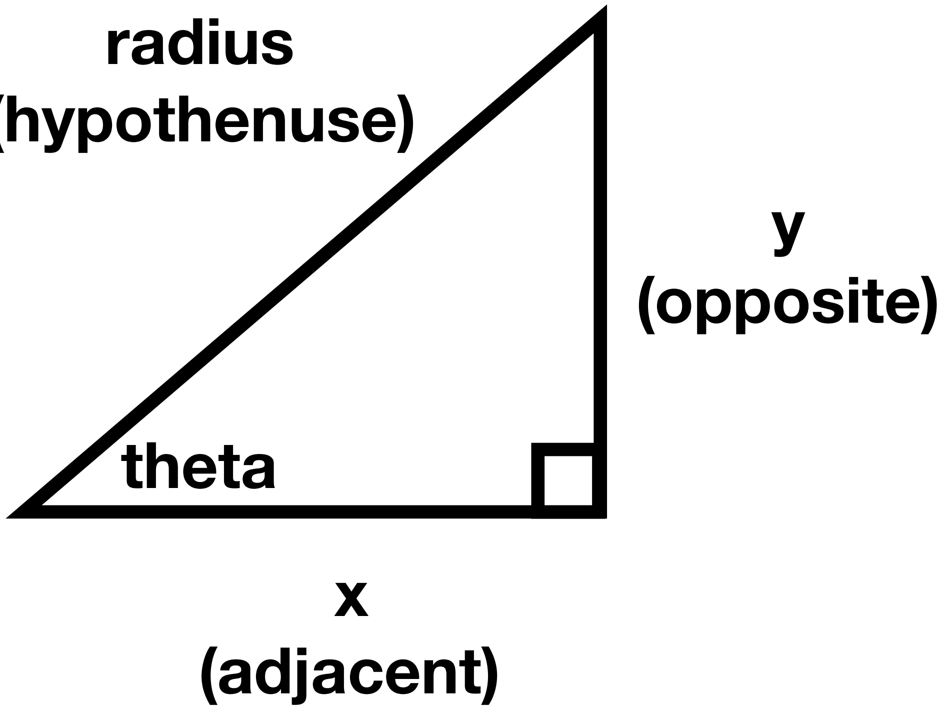
- Word SHOULD NOT overlap other drawn words



```
x, y, index := 0, 0, 1
for dotIsValid( x + width/2, y + height/2) {
        ...
        calibrationX := fixed.Int26_6((x + width/2) * 64)
        calibrationY := fixed.Int26_6((y + height/2) * 64)
        bound = calibrateBound(bound, calibrationX, calibrationY)

        if canFitIn(bound) {
                ...
                break
        }
        index++
        x, y = pickADot(index)
}
```

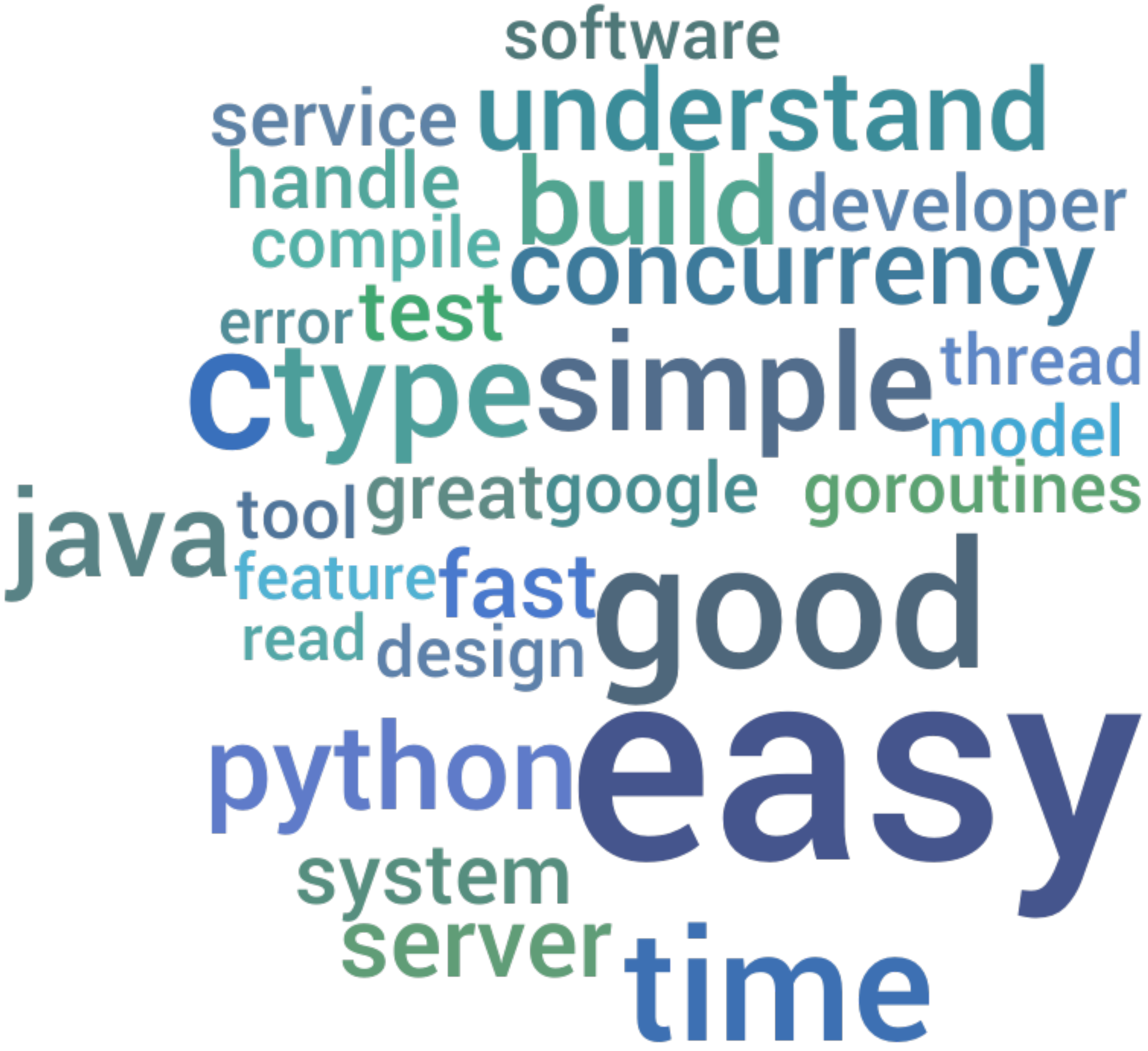
Visualization

7. Create word cloud
   - Word SHOULD NOT overlap other drawn words



```go
func pickADot(i int) (x, y int) {
        index := float64(i)
        radiusIncrement := 0.15
        thetaIncrement := 0.1

        radius := 1.0 + index * radiusIncrement
        theta := 0.0 + index * thetaIncrement

        x = int(radius * math.Cos(theta))
        y = int(radius * math.Sin(theta))

        return x, y
}
```

google


software
service
understand
handle
build
developer
compile
concurrency
error
test
c
type
simple
thread
model
java
tool
great
google
goroutines
feature
fast
good
read
design
python
easy
system
server
time

Tips

- Get started with “A Tour of Go”
- Identify and focus on an interesting project
- Turn to Google and Stack Overflow
- Ask your friends and colleagues

“The secret of getting ahead is getting started.”

-Mark Twain