# Class Diagrams:

**Class Scripture:**

* Attributes:
  + \_words(): list<Word>
  + \_reference: string
  + \_wordsIndex: List<int>
* Constructor:
  + Scripture(string reference, string words)
* Behaviors:
  + BuildWords(string words): void
  + Hidewords(): void
  + GetScriptureString(): string
  + RandomNumber: int
  + AllHidden(): bool

**Class Reference:**

* Attributes:
  + \_book:string
  + \_chapter: string
  + \_verseStart: string
  + \_verseEnd: string
* Constructors:
  + Reference(string: reference)
  + Reference(string: book, string: chapter, string: verse)
  + Reference(string: book, string: chapter, string: verse, string: verseEnd)
* Behavior:
  + GetReference(): string
  + ParseReferenceString(): void

**Class Word:**

* Attributes:
  + \_word: string
  + \_hiddenWord: string
  + \_isHIdden: bool
* Constructor:
  + Word(string: word)
* Behaviors:
  + BuildHiddenWord(): string
  + GetWord(): string
  + HideWord(): void
  + Ishidden(): bool

# Visual Interactions

Class Scripture:
Attributes:
_words(): list<Word>
_reference: string
_wordsIndex: List<int>
Constructor:
Scripture(string reference, string words)
Behaviors:
BuildWords(string words): void
Hidewords(): void
GetScriptureString(): string
RandomNumber: int
AllHidden(): bool

Class Reference:
Attributes:
_book:string
_chapter: string
_verseStart: string
_verseEnd: string
Constructors:
Reference(string: reference)
Reference(string: book, string: chapter, string: verse)
Reference(string: book, string: chapter, string: verse, string: verseEnd)
Behavior:
GetReference(): string   
ParseReferenceString(): void

Class Word:
Attributes:
_word: string
_hiddenWord: string
_isHIdden: bool
Constructor:
Word(string: word)
Behaviors:
BuildHiddenWord(): string
GetWord(): string
HideWord(): void
Ishidden(): bool
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings
Ink Drawings


# Full Description of Classes:

**Class Scripture:**

* Attributes:
  + \_words(): list<Word>
    - Private List that stores each Word class in order
  + \_reference: string
    - Private String that stores the full reference (book Ch:verse(s))
  + \_wordsIndex: List<int>
    - Private List of integers. Used to keep track of what words have been hidden already. As a word is hidden that Words index is removed from the list.
* Constructor:
  + Scripture(string reference, string words)
    - Reference and Words (the verse) are fed to this from main program.
    - This then creates a Reference instance (Reference class takes care of parsing and storing)
    - Calls Reference.GetReference()
    - This then builds the \_words List. Calls BuildWords()
* Behaviors:
  + Private: BuildWords(string words): void
    - Called from Scripture Constructor
    - Parses through string and creates a Word class for each and stores in \_words.
    - Also initiates \_wordsIndex size based on the string.
  + Public: HideWords(): void
    - Called from Main
    - Calls RandomNumber to get index to hide
    - Calls Word.HideWord() to hide that word.
  + Public: GetScriptureString(): string
    - Called from Main
    - Returns a string version of the reference + words. This is what gets printed to screen.
      * Calls Words.GetWord() for each word and returns either the word or the hidden word.
  + Private: RandomNumber: int
    - Private Called from HideWords
      * Randomly selects from list of numbers (\_wordsIndex).
  + Public: AllHidden(): bool
    - Called by Main
    - Iterates through the \_words list and calls Words.IsHidden() for each. If any are not hidden, returns false. Else, returns true
    - Will be used to auto-quit.

**Class Reference:**

* Attributes:
  + \_book:string
    - Private string storing Book
  + \_chapter: string
    - Private string storing Chapter
  + \_verseStart: string
    - Private string storing 1st Verse
  + \_verseEnd: string
    - Private string storing 2nd Verse
* Constructors:
  + Reference(string: reference)
    - Created from Scripture Constructor
      * Will feed the reference string to this
    - Calls ParseReferenceString() to parse it out and assign the Attributes
  + Reference(string: book, string: chapter, string: verse)
    - Looking ahead - will be used if I pull from CSV and have the attributes broken out already (don't need Reference to parse it for me)
  + Reference(string: book, string: chapter, string: verse, string: verseEnd)
    - Looking ahead - will be used if I pull from CSV and have the attributes broken out already (don't need Reference to parse it for me)
* Behavior:
  + Public: GetReference(): string
    - Called from Scripture
    - Returns a string with the appropriate Book, Chapter, Verse(s). Accounts for case if 1 verse or 2.
  + Private: ParseReferenceString(): void
    - Parses the reference string into book, chapter, verse(s)
    - Accounts for the cases of:
      * 1 Nephi OR John
      * 3:16 OR 3:16-17

**Class Word:**

* Attributes:
  + \_word: string
    - Private string storing the word
  + \_hiddenWord: string
    - Private string storing the hidden version of the word
  + \_isHIdden: bool
    - Private boolean tracking if the word is hidden or not
* Constructor:
  + Word(string: word)
    - Takes the string and calls BuildHiddenWord
    - This way as soon as it's initiated, both versions of the word exist.
      * Looking ahead, I want to be able to 'unhide' words too. As an extra functionality - something like "space bar" to temporarily show the words again…
* Behaviors:
  + Private: BuildHiddenWord(): string
    - Called by Word Constructor
    - Simply steps through and creates a hidden word with \_ characters for each letter
  + Public: GetWord(): string
    - Called by Scripture Class
    - Returns the appropriate version of the word string depending on boolean state of \_isHidden
  + Public: HideWord(): void
    - Called by Scripture.HideWords()
    - Simply flips the boolean to true
  + Ishidden(): bool
    - Called by Scripture.AllHidden()
    - Simply reports back state of \_isHidden
    - Put into a method to keep \_isHidden private.