Extension Bundle

Func decode<T: Decodable> (

* T = generic
  + Place holder
  + Anything as long as it conforms to codable

\_ file: String

* \_ file = looks for file
  + inside the bundle

As type: T.Type = T.self

* As Type: T.Type = specificies the Type that T should be decoded
  + If T = Int then T.Type should be decoded as Int
* T.self = default value
  + It will use the same type as generic T
  + All instances of T
  + All things that conform to T
    - Ex Int.self = general all Ints

dataDecodingStrategy: JSONDecoder.DateDecodingStrategy = . deferredToDate

* Must specify custom date format if not use default

keyDecodingStrategy: JSONDecoder.KeyDeocdingStrategy- .useDefaultKeys

* Specify if you want a custom key format if not use default

) -> T {

Guard let url = self.url(forResource: file, withExtension: nil) else

fatalError(“failed to locate \(file) in bundle.”)

* If the file cannot be found in the bundle crash the program

Guard let data = try? Data(contentsOf: url) else

fatalError(“Fail to load \(file) from bundle.”)

* If the file unable to be decoded crash the program

Let decoder = JSONDecoder()

Decoder.dateDecodingStrategy = dateDecodingStrategy

* Date decoding from above
* Default

Decoder.keyDecodingStrategy = keyDecodingStrategy

* Key strategy for data
* Default

Do

Return try decoder.decode(T.self, from: data)

* Returns the data that was decoded

Catch various errors

* Checks for errors before sending the data back
* .keyNotFound
* .typeMismatch
* .valueNotFound
* .dataCorrupted
* Failed to load