C#; cours 4

Développer c’est:

* Être un élément d’une grande chaîne → chacun doit faire un effort
  + Tordre le dev dans la direction dont il a besoin
  + But final est de faciliter la vie du dev de la prochaine étape

Interface vs abstract class → “vous n’habitez pas dans votre contrat de bail.”

* On ne peut pas instancier une interface
* Si on fait une classe c’est pour définir un comportement plus en amont
* On choisit interface car on se réserve le droit de filiation (un seul héritage de classe, autant d’interfaces que l’on veut)

Utiliser une interface que je ne peux pas instancier:

* Créer ce que je veux à travers un object qui l’instancie
* Ex: ICompare
* Class ArtPieceComparer : ICompare { int IComparer.Compare(object x, object y) { return (x as artPiece).title.compareTo((y as ArtPiece).title);}} → Classe à faire en internet pour éviter d’avoir un scope qui fait un non-sens (nested class)

Type casting:

* En cas d’erreur:
* (Type) object → Throw exception
* Object as Type → return null

Boxing → transporter un objet dans un conteneur

* Pour pouvoir utiliser l’interface
* Utilise beaucoup de temps et de mémoire
* Utiliser la généricité → plus explicite et plus efficace
  + Fera le boxing au compile et pas au runtime
  + Utilisation de collections.Generic
  + + offre une réduction du spectre d’erreur de cast

## Structured error handling

En C :

* Valeurs de retour as control:
* return 0 quand tout va bien
* return -1 en cas d’erreurs
* Stockage de l’erreur dans une variable globale → errno
  + Exploitable avec “NOK what?”

Exceptions:

No more return codes, isolate error handling

Use HSE with try, catch, finally, thow

System.Exception e, permet de récupérer au plus haut niveau les exceptions

System.type...., pour permettre de faire un cas précis

Context saving with registered stack state for chained handlers

# XML

* Originates from SGML. → XML = version simplifiée
* eXtensible Markup Language, data description en exchange
* Structured tree graph with <tags>

Ability to serialize any object in an xml file, and then deserialize it

* Serialization example:

Public void serializeXML()

{

XmlSerializer xs = new XmlSerializer(typeof(List<ArtPiece>));

Using

}