* Controllare nel jsx compiler prima di emettere la import che non sia già stata importata
* Capire il modo di come fare riferimento da un template child al parametro lambda del parent
* Group binding expressions in a namespace
* Nel template builder migliorare la clear assicurandosi che vada in deep in tutti i child builders
* Expressioni che sono oggetti nel compilatore vanno messe tra “()“
* Il metodo component deve poter accettare classi e funzioni che implementano behavoir (possibile compilation time?)
* Implementare l’attributo style che accetta un oggetto con tutti gli stili
* Attenzione alle subscription quando si crea un componente, primo o poi vanno pulite
* Valutare di cashare il template builder associato ad un modello e riutilizzarlo al secondo giro
* Ne behavoir, chamare l’attach solo a componente costruito, implementare la detach
* Nei componenti, implementare i child con la seguente logica:
  + Stringa => singolo content di tipo text
* Cambiare la logica di binding come segue:
  + Evitare di passare il proxy a funzioni e metodi
  + Usare il proxy come builder di espressione, poi da valutare e controllare in seguito
  + Evitare quindi, di unbindare / re-bindare tutto
  + Normalizzare il namespace per tutte le operazioni di biding
  + Valuatare se le espressioni di binding debbano implementare il binding a runtime o essere usato solo come segnaposto
  + Valutare se estrarre i metadati di binding / validazione dal file TS e generare una export nel componente con questi metadati o una prorietà sulla classe
  + Implementare il property name
  + Evitare l’uso diretto di parent e use, usare le funzioni di supporto
  + Passare l’espressione padre al proxy
  + Valutare se proxare i metodi
* Supporto “temi” che impostano il valore di default delle proprietà. Da vedere se unica struttura globale, o locale nel componente o entrambe
* Passare il componente padre al figlio
* Implementare l’operatore switch
* Implementare jsx dentro attributi e operatore spread