Progetto Type Script
per il master in Front End Developer
di Luca Radatti

Requisiti Progetto

Questo progetto ha lo scopo di sviluppare un sistema in TypeScript che modella la struttura operativa di un brand di beachwear in plastica riciclata, focalizzandosi sulle interazioni tra clienti, prodotti beachwear e processi di produzione sostenibile.

Attraverso la definizione di interfacce e classi che rappresentano i vari componenti del sistema (IProdotto, ICliente, IProcessoProduzione), il progetto dimostra come la tecnologia possa supportare l'innovazione sostenibile nel settore della moda.
Il progetto richiede solo la scrittura del codice in TypeScript.

Definizione delle interface

```
interface IProdotto {
 tipo: string;
 ID: string;
 tg: string;
 colore: string;
 stato: string;
 assegnaCliente(cliente: ICliente):
void;
}
```

```
interface ICliente {
 nome: string;
 cognome: string;
 email: string;
 metodoDiPagamento: string;
 ordinaProdotto(prodotto:
IProdotto): void;
}
```

```
interface IProcessoProduzione {
 nome: string;
 descrizione: string;
 prodottiInProduzione:
IProdotto[];
 aggiungiProdotto(prodotto:
IProdotto): void;
}
```

Implementazione della classe Prodotto

```
class Prodotto implements IProdotto {
 tipo: string;
 ID: string;
 tg: string;
 colore: string;
 stato: string;

 constructor(
 tipo: string,
 ID: string,
 tg: string,
 colore: string,
 stato: string
 ) {
 this.tipo = tipo;
 this.ID = ID;
 this.tg = tg;
 this.colore = colore;
 this.stato = stato;
 }
```

```
 assegnaCliente(cliente: ICliente): void {
  if (this.stato === "Disponibile") {
  console.log(
  ` Il cliente ${cliente.nome}
 ${cliente.cognome} ha ordinato il prodotto
 ${this.tipo} ${this.colore} con id ${this.ID},
 pagerà con il metodo di pagamento
 ${cliente.metodoDiPagamento}`
  );
  this.stato = "Ordinato";
  }
 }
```

Implementazione della classe Cliente

```
class Cliente implements ICliente {
 nome: string;
 cognome: string;
 email: string;
 metodoDiPagamento: string;

 constructor(
 nome: string,
 cognome: string,
 email: string,
 metodoDiPagamento: string
 ) {
 this.nome = nome;
 this.cognome = cognome;
 this.email = email;
 this.metodoDiPagamento =
metodoDiPagamento;
 }

 ordinaProdotto(prodotto: IProdotto): void
{
 if (prodotto.stato === "Disponibile") {
 console.log(
 ` Il prodotto ${prodotto.tipo}
${prodotto.colore} è stato ordinato!`
 );
 } else {
 console.log(
 ` Il prodotto ${prodotto.tipo}
${prodotto.colore} non puo essere ordinato
perché esaurito :( `
 );
 }
 }
}
```

Implementazione della classe Processo Produzione

```
class ProcessoProduzione implements IProcessoProduzione {
 nome: string;
 descrizione: string;
 prodottiInProduzione: IProdotto[];

 constructor(
 nome: string,
 descrizione: string,
 prodottiInProduzione: IProdotto[]
 ) {
 this.nome = nome;
 this.descrizione = descrizione;
 this.prodottiInProduzione = prodottiInProduzione;
 }

 aggiungiProdotto(prodotto: IProdotto): void {

this.prodottiInProduzione.push(prodotto);
 console.log(
 ` Il prodotto ${prodotto.tipo} ${prodotto.colore} con id ${prodotto.ID} è stato aggiunto al processo di produzione "${this.nome}"`
 );
 }
}
```

Logica di collegamento

Spiegazione dei metodi implementati nelle 3 classi:

Il metodo **assegnaCliente**, quando viene invocato, la prima cosa che fa è controllare se lo **stato del prodotto** è impostato su **disponibile**. Se questo risulta **vero**, manda in **console** il messaggio di **avvenuto ordine**, con i seguenti dettagli:

- **Nome e cognome** del cliente che ha ordinato
- **Tipo**, **colore e ID** del prodotto ordinato
- **Metodo di pagamento preferito** del cliente

Logica di collegamento

Spiegazione dei metodi implementati nelle 3 classi:

Il metodo **ordinaProdotto** esegue anch'esso un controllo dello **stato del prodotto**. Se il prodotto risulta **disponibile**, manda in **console** un messaggio che avvisa il cliente che il **prodotto è stato ordinato**.

Altrimenti, se il controllo dà **esito negativo** e quindi il prodotto risulta **esaurito**, manda in **console** un messaggio che avvisa il cliente che quel determinato **prodotto non può essere ordinato** perché **esaurito**.

Logica di collegamento

Spiegazione dei metodi implementati nelle 3 classi:

Il metodo **aggiungiProdotto** inserisce un **nuovo prodotto** nella proprietà **prodottiInProduzione** di un determinato **processo di produzione** dell'azienda. Questo è possibile grazie al metodo .**push()** e al fatto che la proprietà **prodottiInProduzione** sia un **array**.

Quando il **prodotto** viene **aggiunto**, il metodo manda in **console** un messaggio che indica che il **prodotto specificato** è stato **aggiunto** al **processo di produzione scelto**.

Istanziare i prodotti

Per quanto riguarda i **prodotti**, sono voluto uscire un po' fuori dalle **indicazioni del progetto**, che richiedeva:

*"Le **linee** sono per **Uomo** e **Donna** e sono disponibili vari **modelli**, tra cui "**Relax**" per una normale giornata al mare, "**Active**" per i più sportivi e attivi, e "**Extreme**" per attività a livello professionale come nuoto e surf. Entro il prossimo anno, sarà lanciata una **linea Kids**."*

Ho preferito **non classificare** le **linee dei prodotti** su **Uomo** e **Donna**, ma applicare **l'obiettivo ONU** "**Gender Equality**", evitando così una suddivisione basata sul **sesso biologico**.

I prodotti sono :

- Costume Snorkeling
- Costume Surfing
- Pareo
- Cappello con visiera

I costumi possono essere di 3 modelli:

- Pantaloncino
- Bikini
- Intero

I **pareo e Cappelli** hanno solo un modello con diversi colori e taglie disponibili.

Istanziare i clienti

Ho istanziato **quattro clienti**:

- cliente001
- cliente002
- cliente003
- cliente004

Ogni **cliente** è identificato da un **codice di 3 cifre** con prefisso “**cliente**”, a cui vengono associati i **dati**.

Ad esempio, il **cliente001** è il cliente **Luca Bianchi**, che ha **un'email**: lucabianchi@email.com, ed il suo **metodo di pagamento preferito**, ovvero: **PayPal**.

Istanziare i processi di produzione

Ho istanziato **due processi** di produzione:

- **“In un mare di plastica”**
- **“Meglio vintage che mai”**

Il **processo** di produzione “**In un mare di plastica**” ha come **descrizione**:

“Produzione su commessa e filiera eco-sostenibile sono gli aspetti che caratterizzano il processo produttivo dei costumi Sunnee. Costumi realizzati con poliestere riciclato procurato della plastica trovata sulle nostre spiagge e nei nostri mari. Solo i costumi richiesti dai clienti vengono realizzati, senza eccedenze di magazzino che creano altro inquinamento.”

Il **processo** di produzione “**Meglio vintage che mai**” ha come **descrizione**:

“Recuperiamo capi d'abbigliamento buttati via e lavoriamo i loro tessuti per dargli nuovi aspetti. Non aggiungiamo nessun materiale nuovo ma ricicliamo quello che a qualcuno non serviva più per darlo a chi ne avrà bisogno.”

Test della logica ordini e aggiunta prodotti ai processi di produzione

cliente003.ordinaProdotto(snorkelingSmeraldoShorts);
snorkelingSmeraldoShorts.assegnaCliente(cliente003);

[Log cliente003.ordinaProdotto]
Il prodotto Costume Snorkeling Pantaloncino Smeraldo è stato ordinato!
[Log snorkelingSmeraldoShorts.assegnaCliente]
Il cliente Alessandro Vitale ha ordinato il prodotto Costume Snorkeling Pantaloncino Smeraldo con id S001, pagherà con il metodo di pagamento Carta di Credito

Test della logica ordini e aggiunta prodotti ai processi di produzione

cliente004.ordinaProdotto(cappelloTurchese);
cappelloTurchese.assegnaCliente(cliente004);

[Log cliente004.ordinaProdotto]
Il prodotto Cappello con Visiera Turchese è stato ordinato!
[Log cappelloTurchese.assegnaCliente]
Il cliente Elisa Tosse ha ordinato il prodotto Cappello con Visiera Turchese con id C006, pagherà con il metodo di pagamento PayPal

Test della logica ordini e aggiunta prodotti ai processi di produzione

cliente002.ordinaProdotto(cappelloTurchese);
cappelloTurchese.assegnaCliente(cliente002);

[Log cliente002.ordinaProdotto]
Il prodotto Cappello con Visiera Turchese non puo essere ordinato perché esaurito :(
[Log cappelloTurchese.assegnaCliente]
Il Log non viene visualizzato poiché quando il metodo .assegna cliente viene chiamato lo stato del prodotto è impostato su esaurito.

Test della logica ordini e aggiunta prodotti ai processi di produzione

```
let playKidsUnisex: Prodotto = new Prodotto(
 "Costume da bambino unisex",
 "K001",
 "XS",
 "Nemo",
 "Disponibile"
);
```

Istanziamo un nuovo prodotto

```
inUnMareDiPlastica.aggiungiProdotto(playKidsUnisex);
```

Lo aggiungiamo al processo produttivo

[Log] inUnMareDiPlastica.aggiungiProdotto]

Il prodotto Costume da bambino unisex Nemo con id K001 è stato aggiunto al processo di produzione "In un mare di plastica"

Grazie per l'attenzione

Lascio qui sotto il link per il repository GitHub e quello per provare il codice su CodePen

https://github.com/radattiluca/Project-Type-Script.git

https://codepen.io/Luca-Radatti/pen/vEBPmgM

Progetto Type Script