

ΔΙΑΔΙΚΑΣΙΑ ΔΗΜΙΟΥΡΓΙΑΣ ΝΕΩΝ BLOCK

Η δημιουργία νέων μπλοκ γίνεται με δύο τρόπους:

- 1) Μέσω των εντολών του angular cli και στη συνέχεια τροποποίηση των απαραίτητων services για εμφάνιση του μπλοκ στη βιβλιοθήκη και rendering
- 2) Μέσω της χρήσης του python script το οποίο αυτοματοποιεί τη διαδικασία (ωστόσο είναι καλό να υπάρχει μία γενική αντίληψη της πρώτης μεθόδου σε περίπτωση σφάλματος)

1) MANUAL ΔΙΑΔΙΚΑΣΙΑ

a)

Επιλέγουμε το φάκελο της ομάδας των μπλοκ μέσα στην οποία θα δημιουργήσουμε το νέο μπλοκ και με δεξί κλικ επιλέγουμε να ανοίξει νέο τερματικό σε αυτό το φάκελο (open new integrated terminal)

b)

Εισάγουμε την εντολή :

ng generate component name-of-component

ή για συντομογραφία

ng g c name-of-component

Με αυτό τον τρόπο θα δημιουργηθεί ένα component με όνομα κλάσης **NameOfComponentComponent** και selector **pl-name-of-component**

Παράδειγμα: Θέλω να δημιουργήσω ένα block το οποίο εμφανίζει κείμενα με animation

Η εντολή μπορεί να είναι **ng g c text-animation-one**

Αυτό θα φτιάξει ένα block το οποίο θα έχει όνομα κλάσης **TextAnimationOneComponent** και selector **pl-text-animation-one**

c)

Στη συνέχεια θα πρέπει να δημιουργήσουμε τη φόρμα παραμετροποίησης του block που φτιάξαμε.

Οπότε κάνουμε ακριβώς την ίδια διαδικασία ανοίγοντας terminal μέσα στο φάκελο του προηγούμενου component που δημιουργήσαμε και τρέχοντας την ίδια εντολή προσθέτοντας στο τέλος του ονόματος το λεκτικό “-mods”.

Παράδειγμα: Στο φάκελο που ονομάστηκε από το σύστημα με την προηγούμενη εντολή text-animation-one ανοίγουμε terminal και τρέχουμε την εντολή **ng g c text-animation-one-mods**.

d)

Σε αυτό το σημείο το cli έχει δημιουργήσει δύο εγγραφές μέσα στο module που ανήκουν τα δύο component στο αρχείο **plano-common-blocks.module.ts** μέσα στην παράμετρο **declarations**.

Αυτές τις δύο εγγραφές πρέπει να τις διαγράψουμε και να αποθηκεύσουμε το αρχείο.

Αυτό συμβαίνει επειδή αρχικά τα blocks είχαν σχεδιαστεί να ανήκουν σε αυτό το module το οποίο διαχειριζόταν όλες τις βιβλιοθήκες , υπηρεσίες και πακέτα που έπρεπε να χρησιμοποιούν τα blocks άλλα στην πορεία του project τα blocks έγιναν standalone το οποίο σημαίνει ότι έγιναν ανεξάρτητα , δεν ανήκουν πλέον στο module αλλά εισάγει το καθένα μόνο του μόνο όσες βιβλιοθήκες , υπηρεσίες και πακέτα χρειάζεται οπότε είναι λάθος να είναι δηλωμένα μέσα στο αρχείο του module.

e)

Αφού έχουμε δημιουργήσει τα δύο blocks πρέπει να τα δηλώσουμε στα τρία services που είναι απαραίτητα:

i) Στο service που διαχειρίζεται τις πληροφορίες του block (**pl-component-info.service.ts**) όπως πχ την εμφανιζόμενη ονομασία του (alias) , την περιγραφή του , την εικόνα preview που θα έχει , σε ποιο group ανηκει (header , footer , body) κα.

Εδώ δηλώνουμε σε ένα switch case το όνομα κλάσης του main component πχ

```
case 'TextAnimationOneComponent': {  
    return  
    {  
        alias:"Animated Κείμενα" ,  
        sector:"body" ,  
        group:"text blocks" ,  
        description:"Περιγραφή",  
        imageURL:"assets/plano-component-images/TextAnimationOneComponent .png",  
        restrictions:null  
    }  
    break;  
}
```

ΠΡΟΣΟΧΗ Το switch case θα πρέπει να τοποθετηθεί μαζί με τα cases που ανήκουν στην ομάδα του Το mods component δεν χρειάζεται κάποια δήλωση σε αυτό το service

ii) Στο service που διαχειρίζεται το που θα εμφανιστεί το μπλοκ στη βιβλιοθήκη του builder καθώς και τη σύνδεση μεταξύ του main block και της φόρμας παραμετροποίησης του (**pl-component-resolver.service.ts**).

Σημείο που θα εμφανιστεί στη βιβλιοθήκη:

```
textModules:  
[  
    //1-005||  
    {id:'#COMP_TAO_0001' , selectorKey:'TextAnimationOneComponent'},  
    {id:'#PTBC-002' , selectorKey:'PlanoTextBlockComponent' },  
    {id:'#018786' , selectorKey:'InfoBlocksComponent' }  
],
```

Σημείο σύνδεσης μεταξύ main block και φόρμας παραμετροποίησης:

```
//TEXT
//2-005|

case 'TextAnimationOneComponent': {
  return TextAnimationOneModsComponent
  break;
}
case 'PlanoTextBlockDefaultComponent': {
  return PlanoTextBlockDefaultModsComponent
  break;
}

case 'PlanoTextBlockComponent': {
  return PlanoTextBlockModsComponent
  break;
}
```

ΠΡΟΣΟΧΗ εδώ βλέπουμε οτι δηλώνουμε και τα δύο μπλοκ

iii) Στο service που διαχειρίζεται τη διαδικασία rendering του block (**block-rendering-dictionary.service.ts**)

```
//TEXT COMPONENTS
//005|
{key:'TextAnimationOneComponent' , func: ()=>from( import('../..'/plano-common-blocks/005-text-
blocks/text-animation-one/text-animation-one.component')).pipe
(map(lazyBarrel=>this.createComponentCustom(lazyBarrel.TextAnimationOneComponent))}},
{key:"InfoBlocksComponent" , func: ()=>from( import('../..'/plano-common-blocks/005-text-blocks/info-
blocks/info-blocks.component')).pipe
(map(lazyBarrel=>this.createComponentCustom(lazyBarrel.InfoBlocksComponent))}},
{key:"PlanoTextBlockComponent" , func: ()=>from( import('../..'/plano-common-blocks/005-text-
blocks/plano-text-block/plano-text-block.component')).pipe
(map(lazyBarrel=>this.createComponentCustom(lazyBarrel.PlanoTextBlockComponent))}},
```

ΠΡΟΣΟΧΗ εδώ βλέπουμε οτι δηλώνουμε μόνο το main block

e) Τέλος δημιουργούμε ένα αρχείο json στο φάκελο του main block το οποίο έχει το όνομα της κλάσης του μπλοκ πχ **TextAnimationOneComponent.json** και ως περιεχόμενο αυτού του αρχείου βάζουμε ένα κενό object {} και το αποθηκεύουμε.

f) Μετά απο αυτές τις ενεργειες το μπλοκ είναι έτοιμο να εμφανιστεί στη βιβλιοθήκη των μπλοκ να επιλεγεί για τοποθέτηση μέσα στη σελίδα και να εμφανιστει. Όμως για να λειτουργεί και να συμπεριέρεται σωστά θα πρέπει να αντιγράψουμε στον κώδικά του στο αρχείο ts τον κώδικα του **prototype-component.ts** που βρίσκεται στην ομάδα (φακελο) **000-prototype-block** και στη συνέχεια να αντικαταστήσουμε τα

- selector: "",
 - templateUrl: "",
 - styleUrls:
-
- όνομα της κλάσης
 - το όνομα του json στο import που αφορά το default configuration

με τα λεκτικά που αφορούν το δικό μας block.

2) ΔΙΑΔΙΚΑΣΙΑ ΜΕ ΧΡΗΣΗ ΤΟΥ PYTHON SCRIPT

1) Ανοίγεις terminal στο φακελο app

2) Τρέχεις το script `python .\plano-block-generator.py 001-header-blocks/header-name "Block Alias " "Block Description "` προφανως αντικαθιστώντας τα απαραίτητα λεκτικά "ονομα φακελου ομαδας" "ονομα μπλοκ" "alias" "περιγραφη"

Πχ `python .\plano-block-generator.py 005-text-blocks/text-animation-one "Κείμενα με animation " "Περιγραφή"`

3) Αντιγράφεις στο δημιουργημένο μπλοκ των κώδικα για το ts αρχείο απο το prototype block που βρίσκεται στην ομάδα (φακελο) 000-prototype-block

4) Αντικαθιστας στον αντεγραμμένο κώδικα στο ts τα:

- selector: "",
 - templateUrl: "",
 - styleUrls:
-
- όνομα της κλάσης
 - το όνομα του json στο import που αφορά το default configuration

ΣΗΜΕΙΩΣΕΙΣ ΣΕ ΠΕΡΙΠΤΩΣΗ ΣΦΑΛΜΑΤΟΣ

ΕΛΕΓΧΕΙΣ ΤΡΙΑ SERVICES ΨΑΧΝΟΝΤΑΣ ΠΑΝΤΑ ΤΟ ΟΝΟΜΑ ΤΗΣ ΚΛΑΣΗΣ ΤΟΥ BLOCK

πχ TextAnimationOneComponent

pl-component-resolver.service.ts

i) πρέπει να υπάρχει στο αρχικό object της βιβλιοθήκης το όνομα της κλάσης σαν selector key

(αν δεν υπάρχει ψάχνουμε την παράμετρο που αναφέρεται στην ομάδα που ανήκει το block και δημιουργούμε μία νέα εγγραφή με ένα τυχαίο id)

ii) πρέπει να υπάρχει ένα switch case με το όνομα της κλάσης του main block για να ανοίγει το modification form το οποίο κάνει return την κλάση του modification from block

(αν δεν υπάρχει δημιουργούμε)

block-rendering-dictionary.service.ts

πρέπει να υπάρχει μια εγγραφή με το όνομα της κλάσης σαν key (αν δεν υπάρχει ψάχνουμε τα μπλοκ της ομάδας στην οποία ανήκει κανουμε copy ένα από εκείνα και αλλάζουμε τα λεκτικά ώστε να συμφωνούν με το νέο μας μπλοκ)

pl-component-info.service.ts

πρέπει να υπάρχει ένα switch case με το όνομα της κλάσης

(αν δεν υπάρχει ψάχνουμε τα μπλοκ της ομάδας στην οποία ανήκει κανουμε copy ένα από εκείνα και αλλάζουμε το όνομα της κλάσης και τις πληροφορίες)