9/1/2022

Abdurrahman Gamal Ahmed Gaber

no campany

Clean code

**Ch3 Functions**



Table of Contents

[1-Small : 3](#_Toc124351853)

[1.1-Blocks and Indenting : 4](#_Toc124351854)

[2-Do One Thing: 4](#_Toc124351855)

[2.1- Sections within Functions: 6](#_Toc124351856)

[3-One Level of Abstraction per Function : 6](#_Toc124351857)

[3.1- Reading Code from Top to Bottom: The Stepdown Rule: 7](#_Toc124351858)

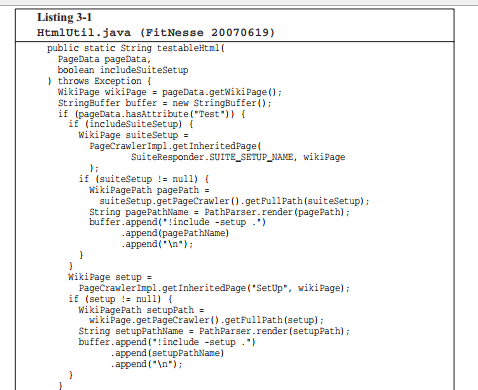
[4- Sections within Functions: 9](#_Toc124351859)

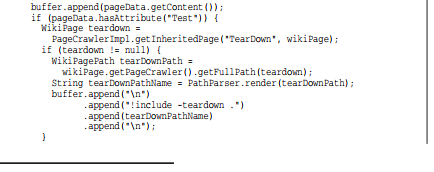
[5- Use Descriptive Names: 10](#_Toc124351860)

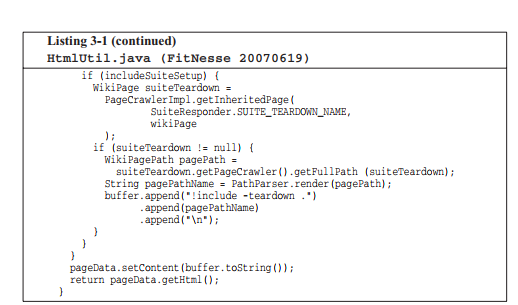
[6- Function Arguments: 10](#_Toc124351861)

الشبتر دة كان مليان معلومات الصراحه . وهو بيتكلم عن ال function وال function دى مهمه جداا عشان بتعتبر من بدهيات التنظيم لاى برنامج

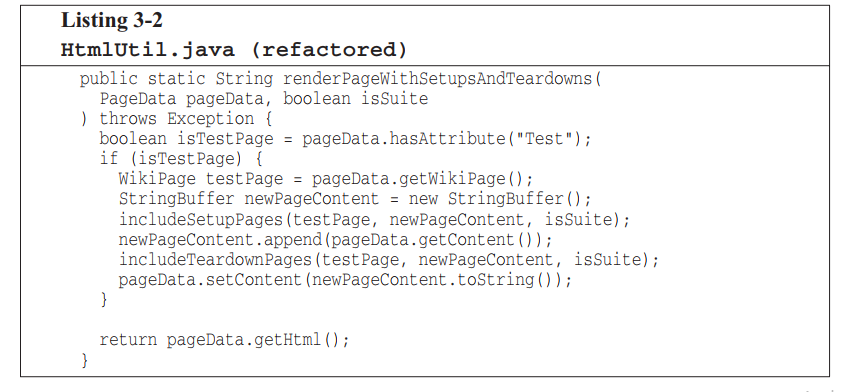
عمنا بوب من البدايه كدة عشان يفهمنا المشكله عطانى اتنين فانكشن عشان نقارن بينهم :دى اول فانكشن







الاسكرين دى باقى الفانكشن الى فوق مش واحده تان

وطبعا باين هى كبيرة قد ايه وبيتحداك انك تفهمها ف تلت دقايق . وحول الفانكسن دى كلها لواحدة تانى كل الى فيها 9 سطور .ودة الشكل للفانكشن بعد التعديل والتحسين

* طبعا بيان الفرق الشاسع بين الاتنين الاوله اصلا مش هحاول افهمها اما التانيه دى لو عرفت السياق الى اتكتبت فيه هفهما بسهوله عادى :والفانكشن الاوله الى مكتوب عليها 3-1 دىكانت معقده بسبب ان فيه level of abstraction ومعنى abstraction هنا ف الشبتر مستوى التعقيد او مستوى التفاصيل الى ف الكود وهنتكلم تانى عن الموضع دة باتفصيل .
* طيب ايه الى خلى الفانكسن التانيه اسهل ف القرايه والفهم ؟

# 1-Small :

القاعدة الاوله عشان اخلى ال function اسهل ف الفهم : انك تخليها small

القاعه التانيه انك تخليها : Small اكتر كمان

وبيقلك ان مفيش قاعدة معينه ممكن تثبتلك ان كل ماكانت ال function اصغر كل مكان احسن . بس على مدار 40 كتب فيهم function كبيرة ممكن توصل ل 3000 سطر ومتوسطه حوال 300 سطر وصغيرة ف حدود 30,20 سطر . والى عمنا بوب اكتسبه من كل التجارب دى ان ال function لازم لازم تبقا صغيرة قدر الامكان .

و فى ال 80s المبرمجين التفقه ان الحجم المناسب لل function هو بحجم الشاشه يعنى اقدر اشوف كل ال function من غير معمل scroll .

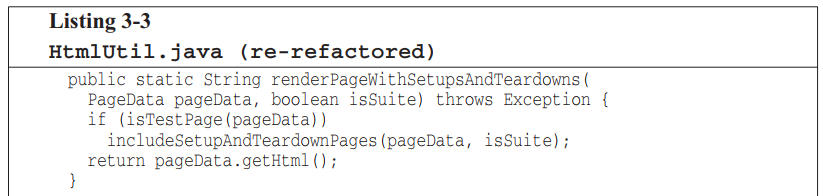
بس الكلام دة طبعا زمان يعنى كان حجم الشاشه ف الوقت دة يشيل 24 سطر اما دلوقتى حجم الشاشه ممكن تشيل 100 سطر ف القاعه بتاعه ال 80s مش صح خالص نطبقها دلوقتى .

وعمنا بوب بيحكلنا ان لما كان ف زياره لصاحبه ف شاف عنده برنامج لما يحرك الموس بينزل نجوم ف البرنامج دة كل ال function الى فيه كان حجمها بين سطرين او تلاته باكتير اوى وكل function لتنقلك لل function الى بعدها .

طيب طول ال function مفروض يبقا قد ايه؟ بيقلك شايف ال function رقم 2-3 ال function بتاعتك لازم تبقا اصغر منها ودة شكل ال function الى عمنا بوب عاوزك توصله.

* وملحوظه 2-3 دة رقم موجود على الاسكرين الى سبتهالك من شويه .
* وكمان الاسكرين الى جايه هى طلعت لما عملنا refactoring لتانى اسكرين الى رقمها 3-2 والاسكرين الى رقمها 3-2 دى ناتجه لما عملنا refactoring الاسكرين رقم 1-3.

ف بختصار الاسكرين رقم 3-3 دى طلعت لما عملنا refactoring للاسكرين رقم 1-3. وطبعا باين الفرق الشاسع ف الحجم .



ممكن تكون بتسئل نفسك هو وصل للشكل دة ازاى ف هو موضحش وساب الاسكرين كدة وبعد شويه هيقلك عمل كدة ازاى كنوع من انواع التشويث يعنى .

## 1.1-Blocks and Indenting :

معنى ال Blocks الى هنا الى هو body بتاع ال if, loop ,switch كدة يعنى .

ف عمنا بوب بيقلك ال Blocks مفروض يبقا سطر واحد بس ومش بس كدة دة السطر دة مفروض يبقا function وفايده ان ال body او ال Blocks يبقا function ان كدة الاسم هيبقا معبر عن الغرض وبالتالى هتفهم الكود بسهوله .

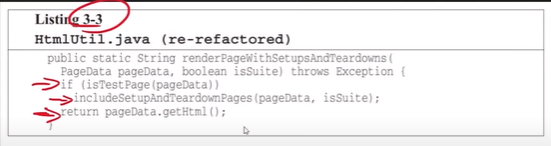
ومفروض تكون فهمت ان مينفعش يكون حجم ال function كبير لدرجه ان يكون فيه nested structures .

# 2-Do One Thing:

بيقلك ان ال function رقم 3-1 بتعمل حجات كتير خالص ودة من اسباب انها معقده وكبيرة .

وال function رقم 3-3 والى كانت صغيرة خالص وواضحه دى كانت بتعمل حاجة واحدة بس والى هى اسم الfunction .

**ف ال function مفروض تعمل حاجة واحدة وتكون بتعملها صح وتعملها هى بس .** ومشكله النصيحه دى انك تعرف ايه الحاجة الواحدة الى الfunction مفروض تعملها .

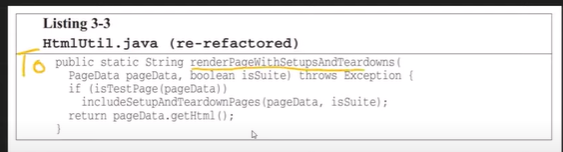
 طيب تفتكر ان الfunction رقم 3-3 بتعمل حاجة واحدة بس؟

لو جينا شفنا ال function هنلاقيها بتحدد

* هل ال pageData هى testPage ولا لا
* لو هى test هتعمل include
* وبعد كدة بتعمل render لصفحه ال html .

طيب كدة افهم انها بتعمل حاجة واحدة ولا لا ؟ الfunction دى بتعمل التلت حجات الى جواها وهم من نفس level ال abstraction يعنى اقدر اوصف ال function دى على انها paragraph . واول ال paragraph يبداء ب to يعنى هضيف to قبل اسم ال function واكمل شرح ليها لو لقيت ان عادى الموضع ماشى بتسلسل ف كدة معناه ان ال function دى بتعمل حاجة واحده .

* تعاله نوصف ال function رقم 3-3



س

**To** renderPageWithSetupsAndTeardowns we check if the pageData is test page and if so we includeSetupAndTeardownPages and in all case we render datapase .

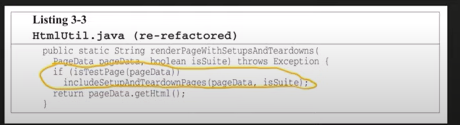
ف انا وصفت الfunction كلها على اكنها one paragraph . ف كدة ال function دى فيها level واحد من ال abstraction وكده هى بتعمل حاجة واحدة بس . والموضع مهم ومن اكتر لمواضيع الى عجبتنى .

**وفى العموما السبب الى بيخلينا نتكتب function اصلا هو اننا بنحلل او بنصغر concept كبير لمجموعه خطوات ف level الabstraction الى بعده او لمجموعه خطوات اقل .**

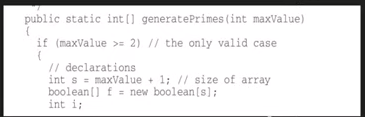
وجه قلك ان صورة 3-1 دى كبيرة عشان فيها اكتر من level of abstraction وصورة 3-2 دى برضو فيها اكتر من level of abstraction والدليل ان قدرنا نعملها refactoring ونطلع منها ب الصورة رقم 3-3.

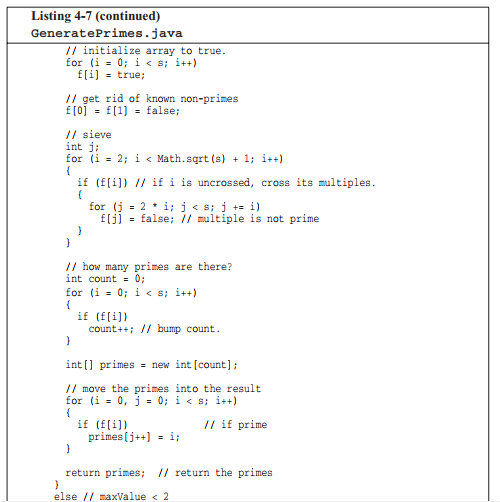
ف عمنا بوب حط قاعدة تانى عشان تعرف ال function دى بتعمل حاجة واحدة وله لا: **قلك لو قدرت تجزء ال function او تصغرها من غير ميبقا الى انت بتعمله اعادة تسميه بس يبقا هى مش بتعمل حاجة واحدة .**

* دى ال function رقم 3-3 واحنا طبقنا عليها قاعدى to وكمان لو حابين نطبق عليها اخر قاعدة هنلاقى منقدرش نصغرها عن كدة ولو قلتلى ممكن احط ال if ف function لوحده هقلك الى بتعمله ملهوش لازم ولا هيفرق اى حاجة ف ال readability .



## 2.1- Sections within Functions:

لو ال function مقسمه لاكتر من section ف كدة ال function دى مش بتعمل حاجة واحده . الاسكرين دى كلها function واحدة والسكشن هنا قصده بيه كلمه declaration و initialize array to true الخ ف هو بيقلك الfunction دى مفروض يحصلها refactoring وتبقا بتعمل حاجة واحدة بس .

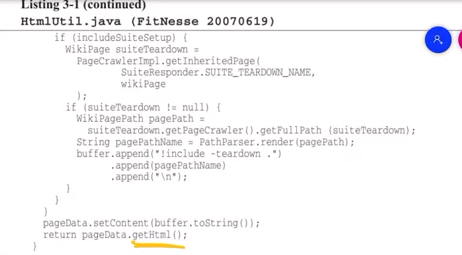


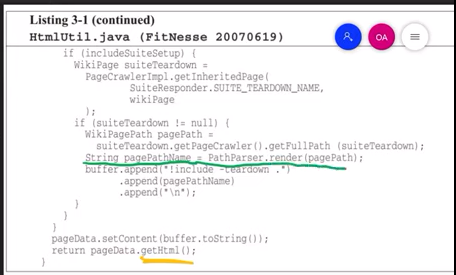
# 3-One Level of Abstraction per Function :

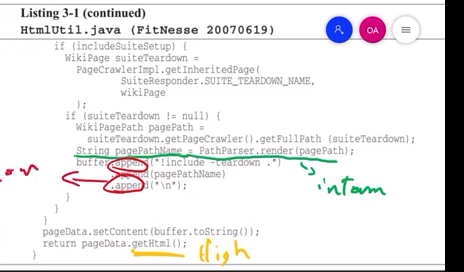
السكشن ده بيتكلم عن ان لاوزم ال function يكون فيها level واحد من abstraction .

وبيوضحلنا كام مثال على موضع ال abstraction من الصورة رقم 3-1 اول صورة عندنا .

* اول حاجة بيقلنا عليها هى ان getHtml دة يعتبر high level of abstraction .



* والجزء دة الى معلم بالخضر دة يعتبر intermediate level of abstraction .
* والجزء الى بالاحمر دة low level of abstraction .

وطبعا ال low level دة واضح ان فيه تفاصيل كتيره.

ف كل ما level ال abstraction قل كل ما التفصيل

بانت.

ف عمنا بوب بيقلك لو لقيت حاجة زى كدة ف دة عك

ومتعملهوش .

## 3.1- Reading Code from Top to Bottom: The Stepdown Rule:

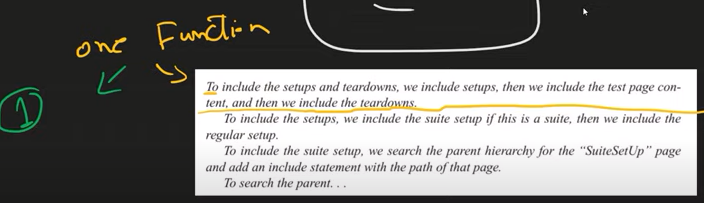
ف السكشن دة عم بوب بيقلنا احنا عوزين الكود يبقا كانه روايه يتقراء من فوق لتحت يعنى اكنك بتقره جرنان او روايه .

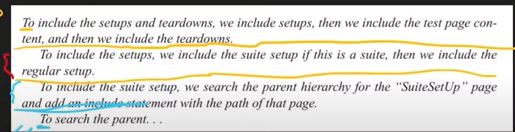
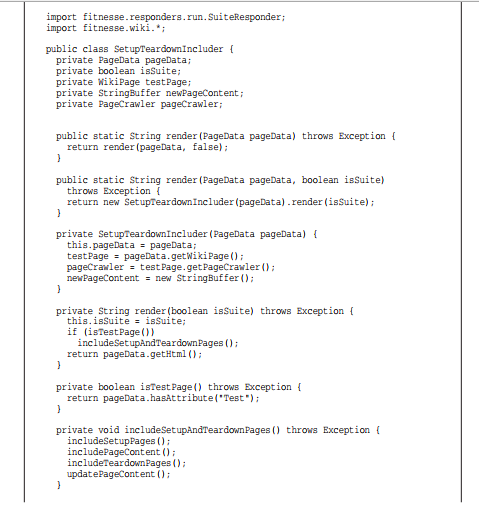
وكل function تكون ال function الى بعدها اقل ف level abstraction . يعنى كل مانزل لتحت ف الكود انقص level abstraction بمقدار واحد بس.

ف الطريقه دى اسمها Stepdown Rule : وبختصار احنا عوزين نحول الكود لparagraphs بدايه كل paragraph كلمه

to وشرحت موضع ال to من شويه .

* وعم بوب بيديك مثال على تحويل الكود لparagraphs وبيقلك ان الكلام الى فوق الخط الاصفر دة بيعبر عن one function وبيكون جوها level واحد من ال abstraction ومينفعش يكون ف اكتر من level .

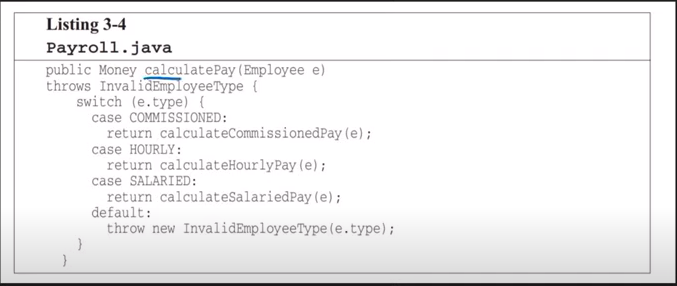


* ف كدة كل function هنمثلها ب paragraph واوله كلمه to ف الاسكرين دى فيها4 paragraph يعنى بتوصف 4 function
* واكيد اموضوع انك تعمل function جواه level واحد بس من ال abstraction دة صعب بس انك تحاول تطبقه هيفيدك . وهو مفتحاح انك تخلى ال function صغيرة وبتعمل حاجة واحدة بس.
* ودة كلاس تطبيق على القاعده . والموضع واضح حتى بالنسبالى انا وانا مش عارف logic وره الكلاس ده .بس انت كل معينك تقع على function تحس انك عارف او فاهم الfunction وحتى مش متعقد منها .
* ف اكيد الوضوع واضح او ازاى كويس وبالاخص لو طبقت قاعه ال to .

# 4- Sections within Functions:

من الصعب قوى انك تعمل switch صغيرة . وهنا لو switch فيها two case بس ف يعتبر كبيرة . وطبعا مينفعش اعمل switch فيها one case عشان ال switch اصلا معموله عشان يبقا فيها اكتر من case .

* ف بكل بساطه نقدر نستخدام ال polymorphism عشان نعمل ال switch يعنى بدال معمل switch هنعمل polymorphism .
* ودة مثال على ازاى استخدام ال polymorphism بدال ال switch وكنت شارح نفس الموضع ف ال design pattern :



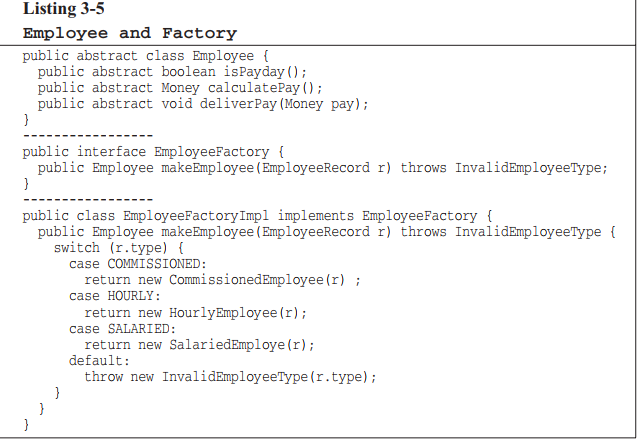
اول حاجة تعاله نشوف مشاكل ال function الى معظمها جاى عشان فيه switch :

1. اول حاجة هى كبيرة
2. لما نضيف نوع موظف جديد ف مفروض ال function دى هفتحها واعدل فيها ف بالتالى هتكبر اكتر .
3. والجزء دة كلامى انا عبد الرحمن : ال function دى بتنتهك مبداء ال principle single responsibility وشرحت الموضع دة ف جزء ال design pattern .
4. وكمان ال function دى بتنتهك مبداء ال open/closed principle عشان لو حصل اى تغير ف الbussines logic هفتح ال function دى واعدل فهيا . ودة كلام عم بوب
5. واسواء عيب ممكن يكون موجود ان يكون ف اكتر من function بنفس الشكل ف العيب دة هو اسواء من كل الى فاته .

ف ممكن يكون فيه function تانيه ف ال system دة اسمها isPay(employee e) و display(employee e) ف بيقلك ال structure ده ضار جداااااا جدااااااااااااااااااااا "لو مش وصلك ليه هو ضار ف شوف شرح DP الى انا عامله"

طيب الحل ايه :

* القاعده الى هنمشى عليها هى لو ال switch هتظهر مرة واحدة بس فى الsystem ف ممكن نتقبلها عادى غير كدة لا .
* وعم بوب بيقول ان هو نفسه سعات مش بيطبقع قاعدة الswitch يعنى كل حاله وليها ظروفها .
* والاسكرين دى بتوضح الحل بس عم بوب مشرحش الاسكرين دلوقتى وهيشرحها ف الاخر متقلقش .



# 5- Use Descriptive Names:

دايما لازم تدى لاختيار الاسماء حقه يعنى متستعجلش وانت بتختار الاسماء ف دة مش تضيع وقت .

ودايما افتكر مبداء واحد اسمه word :" انت بتعرف ان الكود بتاعك clean لما كل حجة بتعملها بتطلع زى ما انت عايز ف دة موشر انك بتكتب ب clean code ف نص طريقه عشان تحقق مبداء word انك تعمل function صغيرة واسمها كويس وبتعمل حاجة واحدة

وكل ماكانت ال function صغيرة كل مكان سهل اختيار اسم معبر ليها ومتخفش من اختيار اسم طويل معبر للfunction والاسم الطويل المعبر احسن من الاسم القصير الى مش معبر او اسم طويل معبر احسن من comment طويل معبر واستخدام ال name convention وقصدى ب ال name convention حاجة زى ال camel case "

* نرجع لعم بوب بيقلك متخفش تاخد واقت ف اختيار الاسم وممكن تجرب اكتر من اسم وتجرب تقره الكود بلاسم لحاد متوصل لاحسن اسم معبر للfunction
* وطبيعى وانت بتبحث عن اسم معبر دى ممكن يادى الى اعاده هيكله الكود .
* خلى اسماءك ثابته يعنى استخدام نفس المصطلحات او نفس ال pattern . يعنى خلى ليك style ف ال naming .

# 6- Function Arguments:

احسن عدد Arguments الى هتخدها function هو صفر يعنى تكون ال مبتخداش Arguments .

* تانى افضل حاجة هى انك تخلى ال function تاخد one Arguments
* تالت افضل حاجة هى انك تخلى ال function تاخد two Arguments
* ومفروض تتجنب قدر المستطاع ان يكون ف function بتاخد Arguments 3
* اكتر من 3 Arguments مفروض يبقا عندك مبررات قويه وبرضو مينفعش تعمل كدة يعنى من الاخر مينفعش تخلى ال function تاخد اكتر من 3 Arguments

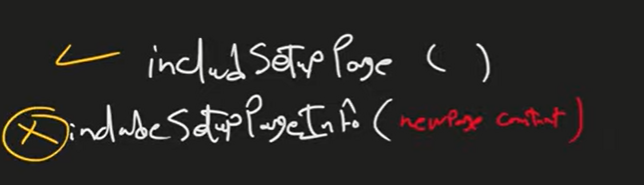
ودة عشان ال Arguments بتاخد معايه جهد زهنى عشان تفهمها .

ولو اخدنا مثال زى : لو خلينا مثلا ال function رقم 2-3 تاخد argument اضافى زى ال buffer ف كدة هيخلى الموضع اصعب للفهم يعنى وانت بتقره الfunction هتكون عاوز تعرف هى بتعمل ايه وكمان عاوز تعرف ايه لزمه الbuffer ف دة جهد اضافى يجب .



ودة مثال تانى : طبعا الاسهل ف الفهم الfunc الاوله ودة بسبب انها مش بتاخد argument

تانى func دى بتاخد argument من abstraction level مختلف عن اسم الfunction وهيجبرك تعرف تفاصيل اكتر مش وقتها دلوقتى .



ومن اهم الاسباب الى بتخلينا نبعد عن ال argument ان ال argument بتخلى موضوع ال testing صعب .يعنى عشان تعمل مجموعه من ال test case تتاكد بيها ان الfunction شغاله مظبوط المضوع هيكون صعب وكبير .

ف لو مفيش اى argument موضع الtesting هيكون سهل خاالص.

اما لو فيه argument واحد موضع الtesting هيكون مش صعب برضو .

اما لو فيه اتنين argument ف موضوع الtesting هيكون صعب شويه .

اكتر من اتنين argument الموضع هيبقا صعب فعلا .

وموضع ال output argument اصعب ف الفهم من ال input argument . يعنى احنا متعودين على فكرة ان الحاجة الى هتدخل ال function هتكون عن طريق ال argument والحاجة الى هتطلع من الfunc هتكون عن طريق ال return value ف دة المتعارف متعودناش ان ف حاجة تخرج من ال function عن طريق argument . عشان كدة ال output argument بتكون حاجة غريبه او مش متعودين عليها .

ف افضل حاجة ان يكون ف argument واحد بعد ان ميكونش فيه اى argument .

## 6.1- Common Monadic Forms :