

“Name carefully- as if a first-born child.”

- Think carefully and don't settle
 - Try different names until you're happy
 - Learn to use built-in IDE refactoring features

```
public class DecoderPenalties
{
    private float insLogProb = LOG_PROB_ZERO;
    private float callLogProb = LOG_PROB_ZERO;
    private float ...

    public float getInsLogProb()
    {
        return insLogProb;
    }
}
```

Enter new name, press Enter to refactor

```
private int numWid
private double co

public void addWi
{
    numWidgets++;
}
```

Refactor

at	Rename...	Ctrl+R, Ctrl+R
	Extract Method...	Ctrl+R, Ctrl+M
	Encapsulate Field...	Ctrl+R, Ctrl+E
	Extract Interface...	Ctrl+R, Ctrl+I
a, b	Remove Parameters...	Ctrl+R, Ctrl+V
a, b	Reorder Parameters...	Ctrl+R, Ctrl+O

“Name carefully- as if a first-born child.”

- Intention-revealing names
 - Why? What? How? Units?

```
int d; // elapsed time in days
```

```
int elapsedTime;
```

```
int elapsedTimeInDays;
```



“Name carefully- as if a first-born child.”

- Avoid disinformation

```
HashSet<Account> accountList;  
  
long accountName;  
  
String numWidgets;
```

```
HashSet<Account> accountGroup;  
  
long accountNumber;
```

```
int a = 1;  
if ( 0 == 1 )  
    a = 01;  
else  
    l = 01;
```

“Name carefully- as if a first-born child.”

- **Make meaningful distinctions**
 - Avoid noise words that add anything
 - Reader should be able to determine difference between similar items

```
int count;  
int theCount;  
int aCount;  
  
double amountVariable;  
String nameString;  
Customer custObject;  
  
getActiveAccount();  
getActiveAccounts();  
getActiveAccountInfo();
```

“Name carefully- as if a first-born child.”

- Use pronounceable names
 - “If you can’t pronounce it, you can’t discuss it without sounding like an idiot”

```
class DtaRcrd102
{
    private Date genymdhms;
    private Date modymdhms;
    private final String pszqint = "102";
}
```

```
class Customer
{
    private Date generationTimestamp;
    private Date modificationTimestamp;
    private final String recordId= "102";
}
```

“Name carefully- as if a first-born child.”

- **Avoid encodings**

- e.g. Hungarian notation, member prefixes

- Has become unnecessary

- Modern IDEs, strongly typed languages, trend towards small classes and methods

```
boolean bBusy;  
int iListSize;  
PhoneNumber strPhone;  
  
private int m_iLastItem;  
m_iLastItem++;
```

```
boolean isBusy;  
int listSize;  
PhoneNumber phone;  
  
private int lastItem;  
lastItem++;  
this.lastItem++;
```

“Name carefully- as if a first-born child.”

- **Be consistent**

- Pick one word per concept

- Use fetch, retrieve or get, not all of them

- If plural, probably should contain multiple things

```
fetchFullPage()  
retrieveHeader()  
getFooter()  
  
setTemp(double newTemp)  
updateTime(double newTime)  
  
double scores;  
int [] score;
```

```
fetchFullPage()  
fetchHeader()  
fetchFooter()  
  
setTemp(double newTemp)  
setTime(double newTime)  
  
double sumScores;  
int [] scores;
```

“Name carefully- as if a first-born child.”

- **Avoid gratuitous context**
 - Don't prefix every class name in a project
 - Use a namespace instead

```
class MailingAddress
{
    String addrState;
    long   addrZip;
}
```

```
class MailingAddress
{
    String state;
    long   zip;
}
```

```
...
String addrState = "MN";
...
```