GitHub - SourceTree - 사용 가이드.

-2016.12.04 - 처음작성.

문 서 명 : GitHub - SourceTree - 사용 가이드.
작 성 자 : bitcamp - kanu
작 성 일 : 2016.12.04
version : 0.1

참고 사이트
http://stackoverflow.com/ - 개발자 추천 사이트.
https://www.gitignore.io/ touch .gitignore 프로젝트 에서 제외할 파일을 알려준다.
https://git-scm.com/book/ko/v2 깃허브 한글 매뉴얼

gitbash down
https://git-scm.com/download/win

참고서적.

[i18n]
commitEncoding = utf-8
logOutputEncoding = utf-8

설치시 이슈 리포트.

- warning: templates not found 오류 발생시 "System git" 으로 설정.
- 한글 깨짐 발생시 설정 파일에 아래 내용 추가.

```
[i18n]
commitEncoding = utf-8
logOutputEncoding = utf-8
```

소프트 웨어 형상 관리(SCM Software Configuration Management) 란

1. 소프트웨어 변경 요구가 발생한 것 부터 구현까지의 전 과정을 제어하고 기록하고 보고하는 변경 관리.
2. 개발주기 동안의 변화하는 코드 와 라이브러리, 관련 문서 등을 저장 하고 관리.
3. 제품의 릴리즈나 빌드에 반영된 변경을 감사하고 관리.

버전관리의 필요성.

1. 여러 개발자가 공동 개발이 가능 하다.
2. 소스 코드의 버전 관리가 가능하다.
    a. 프로젝트 전체를 특정 시간으로 되돌릴수 있다.
    b. 원하는 파일만도 특징 시간으로 코드를 되돌릴수 있다.
3. 누가 어느 부분을 수정 했는지 확인이 가능 하다.

SourceTree Menu

1. 커밋(Commit)
    a. 변경된 파일을 저장소에 제출.
2. 푸시(push)
    a. Commit 한 내용들을 원격 저장소의 master 브랜치에 업로드 한다.
3. 풀

a. 원격저장소의 commit 내용을 로컬 저장소로 가져옵니다.
b. 병합을 같이 수행.

4. 페치(fatch)
   a. 원격저장소의 commit 내용을 로컬 저장소로 가져옵니다.
   b. 커밋 내용을 확인한 후 수동으로 병합 해야함. (이방식을 추천)

5. 브랜치 - 중요한 개념이나 나중에.
6. 태그 - 중요한 개념이나 나중에.
7. 병합

협업도구.

1. 이슈 트래커
   a. 버그보고 ,기능개선 건의, 그 외 프로젝트에 관련된 주제(이슈)를 등록할수 있는 공간
   b. 메뉴 구성
      i. 담당자: 이슈 담당자 지정가능.
      ii. 알림:@<name>형식으로 특정 그룹이나 특정 사용자에게 알림.
      iii. 라벨: 카테고리 역활의 라벨지정 기능.
      iv. 커밋 레퍼런스 : 커밋 해시를 써두면 자동으로 해당 커밋에 링크.
      v. 마일스톤: 이슈 그룹으로 만드는 표식을 지정.
      http://blog.naver.com/PostView.nhn?blogId=softpro&logNo=130011650118
      (게임 개발 관점의 마일스톤 참고)
      [추가설명]
      마일스톤(milestone)이란 프로젝트 진행 과정에서 특기할 만한 사건이나 이정표를 말한다. 예를 들어, 프로젝트 계약, 착수, 인력투입, 선금 수령, 중간보고, 감리, 종료, 잔금 수령 등 프로젝트 성공을 위해 반드시 거쳐야 하는 중요한 지점을 말한다.[1]

      마일스톤은 프로젝트 일정관리를 위해 반드시 필요한 지점을 체크하기 위해 사용한다. 프로젝트 성공을 위해 필수적인 사항들을 각 단계별로 체크함으로써 전체적인 일정이 늦춰지지 않고 제 시간 안에 과업이 종료될 수 있도록 관리하는데 도움을 준다. 다만, 마일스톤은 프로젝트 진행 과정에서 결정적으로 중요한 핵심적인 사항들만 체크하기 때문에, 그다지 중요하지는 않더라도 프로젝트 진행에 꼭 필요한 다양한 요소들을 상세하게 파악하기 힘들다는 단점이 있다

2. 위키(wiki)
   위키 페이지를 장성 할수 있다.
3. 풀 리퀘스트
   저장소와 저장소 사이의 병합. (e.g. 포크 한 원격 저장소화 병합)
4. Guthub 에서의 코드 리뷰

협업규칙

1. 커밋 단위
    a. 커밋 하나는 하나의 의도와 이미만을 가져야 합니다. 한번에 여러 파일을 수정하더라고 그 의도는 단 하나여야함, 버그 수정이던 새로운 기능 추가던 적용
2. **커밋 메시지 작성 규칙**
    a. **[category]-[simple message]**
    [detailed description]
    b. **[Category] 작성규칙 예시.**
        i. **[fix] 잘못된 부분 수정**
        ii. **[add] 기능 추가**
        iii. **[mod] 코드 수정**
        iv. **[rm] 기능삭제**
    c. **가이드 라인.**
        i. **왜 커밋했는지 설명.**
        ii. **버그 수정의 경우 원래 어떤 문제가 있었는지 설명**
        iii. **사용중인 이슈 트래커가 있다면 해당 이슈의 하이퍼링크 포함**

C,C++ 개발 표준 가이드.
출처:http://blog.daum.net/naline1213/7592240

개요
10, 15년전 Microsoft의 개발자중 헝가리 사람의 프로그래머가 쓰던 변수 명명법. MS내부에서 따라쓰기 시작하던 것이 점차 전세계의 프로그래머들에게 널리 퍼져 이젠 프로그램 코딩시 변수 명명의 표준적인 관례가 되었다. 그러나 실제로 현장에서 일하다 보면 헝가리안 표기법을 제대로 지키는 개발자는 그리 많지 않다. 어느정도 개발 경험을 가지고 있는 프로그래머는 물론 심지어 시중의 프로그래밍 서적에서 조차 저자마다 변수명을 개인에 따라 가지각색으로 짓고 있어서 처음 프로그램을 배우는 입문자들들이 변수 명명에 대한 기준을 제대로 잡지 못하고 있는 실정이다. 솔직히 필자도 얼마전까지 이런 변수 명명에 대한 관례를 잘 지키지 않았다. 그러나 변수 명명에 관한 표준화된 관례를 지켜주면 코드의 가독성을 높여줄 뿐 아니라 예를 들어 카운터 변수를 *count*라고 지을지 cnt라고 지을지 고민하지 않아도 되는 편리함을 누릴 수 있다.

1. 명명 규칙. (기본설명)

x_xXxxxxxx
0123456789
0 : 변수의 위치를 지정한다. g(전역변수), m(멤버변수), 없음(지역변수)
1 : 0 위치에 g 나 m 을 지정한 경우 _ 을 기술한다.
2 : 자료형의 종류를 나타낸다.
n, i : 정수형(n은 카운트 목적, i는 인덱스 목적)
l : long 형
u : 부호없는 정수형
w : 부호없는 2byte 정수형
dw : 부호없는 4byte 정수형
p : 포인터 타입
f, d : 실수형(f는 float, d는 double)
sz : char 배열(문자열 표현)
클래스 이름에 대해서는 관습적으로 자음축약형을 사용한다.
3 ~ : 변수의 의미 있는 이름을 기술하며, 3 위치는 대문자를 사용한다. 변수 이름이 너무 긴 경우 자음만을 기술한다. 예) g_nCnt

2. 형식 접두사.

형식	접두사	설명	e.g
bool	b		bool bTrue
char	c		char cLetter
int	i		int iNum
long	l		long lNum
float	f		float fPrecent
double	d		double dPrecent
형식	접두사	설명	e.g
static	s		static short ssChoics
array	rg		float rgfTemp[16]
Point	p		int* piAddr
char* (문자열)	sz	문자열	char* szName[16]
함수포인터	pfn	함수포인터	int (*piFnFunc1)(int,int)
struct	t	구조제	
enum	e		
전역변수	g_	전역변수	strng* g_psBuffer
멤버변수	m_	멤버변수	
상수형식(?)	k	constant formal parameter	void vFunc(const long

			klGalaxies)
레퍼런스	r		void vFunc(long &rlGalaxies)
String	str		String strName
동적할당	prg	동적할당 변수	:char *prgGrades;
HANDLE	h	핸들	hWnd
:number, quantity	n	:number, quantity	int nNum
used as size	x/y		int xWidth,yHeigth

3. 명명 예시

```
unsigned char ucByte;          :한 바이트 데이타
         char cChar;           :한 문자
unsigned char rgucByte[10];    :바이트 데이타 10개
         char rgcChar[10];       :문자 데이터 10개
         char szChar[16 +1];   :문자 16개를 저장할 수 있는 문자열 공간
```

4. 명명 규칙의 종류
   a. 파스칼(Pascal) - 구분이 되는 단어를 대문자로 사용 (class,namespace,함수,프로퍼티)
   b. 카멜(Camel) 구분되는 단어는 대문자로 시작하지만 맨 앞자는 소문자. (변수,매개변수)
   c. 헝가리언 표기법