Native App
iOS 개발 가이드

Version 2.0

STC LAB

Surfing the Waves of Change

iOS

iOS Client API는 iPhone 애플리케이션에 넷퍼넬을 적용하는 방법을 알아봅니다.

넷퍼넬 API 적용 시 **64 페이지의 고급 설정** 방법을 참고하여 디버그 모드로 설정 후 진행하세요.

Base Tutorial

NetFunnel.framework 추가

/Library/Release(Debug)-iphoneos(iphonesimulator) 디렉터리에서 알맞은 NetFunnel.framework 를 선택하여 자신의 Project 에 추가하세요.

구분		용도
Debug 개발용	Debug-iphoneos	log 남김, device 용
	Debug-iphonesimulator	log 남김, simulator 용
	Debug-universal	log 남김, device/simulator 통합
Release 배포용	Release-iphoneos	log 없음, device 용
	Release-iphonesimulator	log 없음, simulator 용
	Release-universal	log 없음, device/simulator 통합

****. Release : 배포 버전 (log 를 남기지 않음), Debug: 테스트 버전(log 남김)***

NOTE

Project Properties → General → Linked Frameworks and Libraries → + (add item) → Add Other...

Global Config 설정

넷퍼넬 서버의 IP/포트와 같은 기본 정보를 애플리케이션 실행 시에 초기화하세요.

■ 설정 방법

setGlobalConfigObject 를 이용하여 초기 값 설정

```
01. - (void)initValue:(NSString *)nid
02. {
03.     [NetFunnel setGlobalConfigObject:@"nf2.netfunnel.co.kr" forKey:@"host" withId:nil];
04.     [NetFunnel setGlobalConfigObject:@"https" forKey:@"proto" withId:nil];
05.     [NetFunnel setGlobalConfigObject:@"443" forKey:@"port" withId:nil];
06.     [NetFunnel setGlobalConfigObject:@"service_1" forKey:@"service_id" withId:nil];
07.     [NetFunnel setGlobalConfigObject:@"act_1" forKey:@"action_id" withId:nil];
08. }
```

■ 설정 가능 변수

- host : NetFunnel Server Host/IP 주소
  (Default : NETFUNNEL_DEF_HOST=@"test.netfunnel.co.kr")
- port : NetFunnel Server Port 번호 (Default : NETFUNNEL_DEF_PORT=80)
- proto : 요청 Prototype (http|https) (Default : NETFUNNEL_DEF_ROTO=@"http")
- proto : 요청 URI Query (Default : NETFUNNEL_DEF_QUERY=@"ts.wseq")
- service_id : Service ID (Default : NETFUNNEL_DEF_SERVICE_ID=@"service_1")
- action_id : Action ID (Default : NETFUNNEL_DEF_ACTION_ID=@"act_1")
- conn_timeout : Connection Timeout (Default : NETFUNNEL_DEF_CONN_ TIMEOUT=3.0)
- conn_retry : Connection Retry Count (Default : NETFUNNEL_DEF_CONN_ RETRY=1)
- ipblock_wait_time : IPBlock 시 가상 대기창을 보여주는 시간
  (Default : NETFUNNEL_DEF_IPBLOCK_WAIT_TIME=10.0)
- wait_view_object : Custom 대기창 Object

Delegate 작성

넷퍼넬 요청 후에 서버로부터 전달된 응답 처리를 해주는 Delegate 를 정의해줘야 합니다.

delegate

```
01. @protocol NetFunnelDelegate <NSObject>
02.
03. @required
04. -(void)NetFunnelActionSuccess:(NSString *)nid withResult:(NetFunnelResult *)result;
05. -(void)NetFunnelCompleteSuccess:(NSString *)nid withResult:(NetFunnelResult *)result;
06.
07. @optional
08. -(void)NetFunnelActionContinue:(NSString *)nid withResult:(NetFunnelResult *)result;
09. -(void)NetFunnelActionError:(NSString *)nid withResult:(NetFunnelResult *)result;
10. -(void)NetFunnelActionBlock:(NSString *)nid withResult:(NetFunnelResult *)result;
11. -(void)NetFunnelActionIpBlock:(NSString *)nid withResult:(NetFunnelResult *)result;
12. -(void)NetFunnelAliveNoticeContinue:(NSString *)nid withResult:(NetFunnelResult *)result;
13. -(void)NetFunnelAliveNoticeError:(NSString *)nid withResult:(NetFunnelResult *)result;
14. -(void)NetFunnelAliveNoticeBlock:(NSString *)nid withResult:(NetFunnelResult *)result;
15. -(void)NetFunnelAliveNoticeIpBlock:(NSString *)nid withResult:(NetFunnelResult *)result;
16. -(void)NetFunnelCompleteError:(NSString *)nid withResult:(NetFunnelResult *)result;
17.
18. @end
19.
```

required

-(void)NetFunnelActionSuccess:(NSString *)nid withResult:(NetFunnelResult *)result

// 정상적으로 key(순번)을 발급 받았을 때 호출된다.(or 대기가 종료되는 시점)

{ // 여기에 제어하고자 했던 **Business Logic 을 호출**하면 된다. }

-(void)NetFunnelCompleteSuccess:(NSString *)nid withResult:(NetFunnelResult *)result

{ // key(순번) 반환이 정상 처리되었을 때 호출된다. }

optional

optional delegate 는 선언할 필요가 없으며, 활용할 필요가 있을 경우만 선언하면 된다.

-(BOOL)NetFunnelActionContinue:(NSString *)nid withResult:(NetFunnelResult *)result

{ // 대기가 발생되는 상황에서, TTL 주기마다 반복되어 호출된다.

return YES; // 대기창 표시: YES(default), 대기창 미표시: NO

}

-(void)NetFunnelActionBlock:(NSString *)nid withResult:(NetFunnelResult *)result

{ // 해당 액션 ID 를 block 처리했을 때 호출된다. }

-(void)NetFunnelActionIpBlock:(NSString *)nid withResult:(NetFunnelResult *)result

{ // 액세스 제어 기능을 통해 차단되었을 때 호출된다. }

-(void)NetFunnelAliveNoticeContinue:(NSString *)nid withResult:(NetFunnelResult *)result

{ // alivenotice 가 정상적으로 처리되었을 때 호출된다. (반복) }

-(void)NetFunnelAliveNoticeError:(NSString *)nid withResult:(NetFunnelResult *)result

{ // alivenotice 반복 호출 중 에러가 발생되었을 때 호출된다. }

-(void)NetFunnelAliveNoticeBlock:(NSString *)nid withResult:(NetFunnelResult *)result

{ // alivenotice 반복 호출 중 해당 액션 ID 를 block 처리했을 때 호출된다. }

-(void)NetFunnelAliveNoticeIpBlock:(NSString *)nid withResult:(NetFunnelResult *)result

{ // alivenotice 반복 호출 중 액세스 제어 기능을 통해 차단되었을 때 호출된다. }

-(void)NetFunnelCompleteError:(NSString *)nid withResult:(NetFunnelResult *)result

{ // key(순번) 반환이 정상적으로 처리되지 못했을 때 호출된다. }

Request 전달

■ 요청 Type

» Action (Required)

넷퍼넬 서버에 자원 사용 허가를 받는 요청입니다. Action 요청 시 순번을 부여받게 되며 자신의 순번에 도달하지 않았다면, 사용자 화면에 대기창을 출력하여 대기를 시켜줍니다.

- 기본적인 Action 요청 방법

```
01. [[[NetFunnel alloc] initWithDelegate:self pView:self.view] action];
```

- Service ID 와 Action ID 를 지정해서 요청하는 방법

```
01. [[[NetFunnel alloc] initWithDelegate:self pView:self.view]
        actionWithSid:@"service_1" aid:@"act_1"];
```

- Service ID 와 Action ID 를 지정해서 요청하는 방법, nid 포함

action 중복 사용시 각 action 구분 값 : nid(NSString)

action 에서 nid 지정 시 Complete/Alive 에서도 nid 지정 필수 (쌍을 이루도록 구성)

```
01. [[[NetFunnel alloc] initWithDelegate:self pView:self.view] actionWithNid:@"nidString" config:
        [[NSDictionary alloc]
            initWithObjectsAndKeys:@"service_1",@"service_id",@"act_1",@"action_id", nil]
    ];
```

» Complete (Required)

점유했던 자원을 반환하는 요청으로 Action 을 실행한 후에 자원을 모두 사용한 이후에는 꼭 실행해줘야 합니다. 만약 실행하지 않는다면 불필요한 대기가 발생할 수 있습니다.

- 기본적인 Complete 요청 방법

```
01. [[[NetFunnel alloc] initWithDelegate:self pView:self.view] complete];
```

- Complete 요청 방법, nid 포함

```
01. [[[NetFunnel alloc] initWithDelegate:self pView:self.view]
        completeWithNid:@"nidString"   config:nil]
```

» Alive (Optional)

자원을 점유한 이후에 일정 시간(자원 반환 전까지) 동안 계속 자원을 점유하고자 할 때는 Alive 요청을 해야 합니다. (하나의 transaction 제어로 끝나는 것이 아닌, 여러 단계에 걸쳐 제어하고자 하는 경우에 사용, timeout 으로 빠지지 않도록 해주는 역할)

- 기본적인 Alive 요청 방법

```
01.  [[[NetFunnel alloc] initWithDelegate:self pView:self.view] alive];
```

- Alive 요청 방법, nid 포함

```
01.  [[[NetFunnel alloc] initWithDelegate:self pView:self.view]
         aliveWithNid:@"nidString"    config:nil];
```

대기창 디자인

NetFunnelWaitView Class 와 NetFunnelWaitViewProtocol 을 이용해서 자신만의 대기창을 만들 수 있습니다. (샘플 프로젝트 참조: CustomWaitView.m)

- 커스텀 대기창 적용 방법

```
01.  [NetFunnel setGlobalConfigObject:[[CustomWaitView alloc]
         initWithParentView:self.view]
             forKey:@"wait_view_object" withId:@"nidString"];
```

대기창 없이 backgroud 에서 제어하고자 하는 경우

NetFunnelWaitView Class 와 NetFunnelWaitViewProtocol 을 이용해서 자신만의 대기창을 만들 수 있습니다. (샘플 프로젝트 참조: CustomWaitView.m)

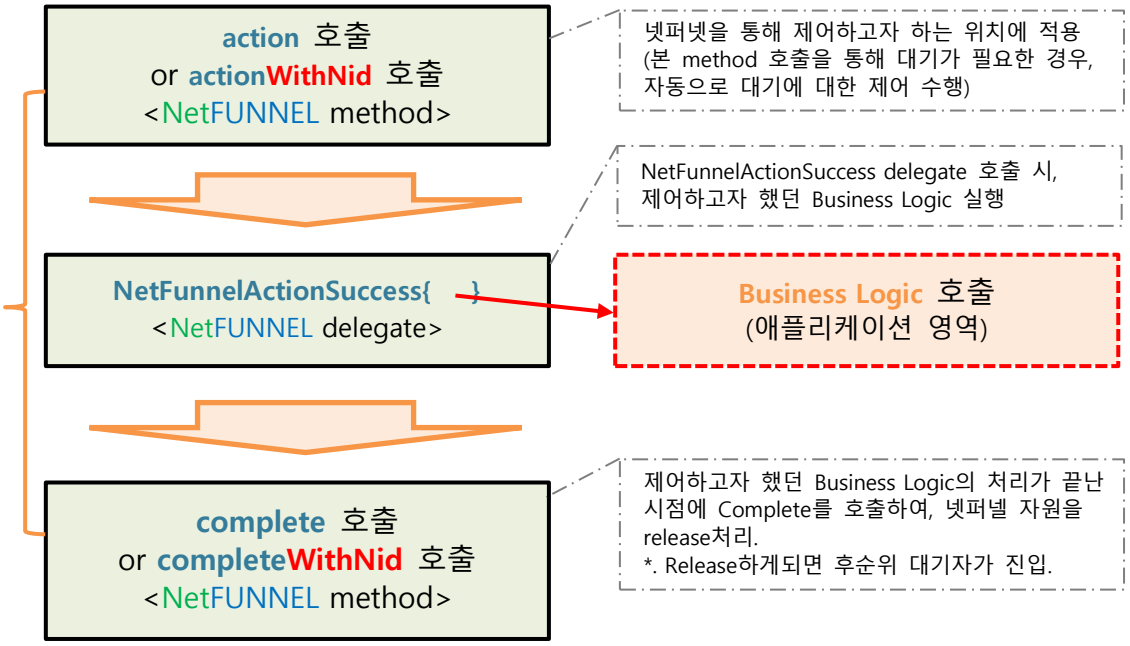
- 대기창을 표시하지 않는 적용 방법

```
01.  -(BOOL)NetFunnelActionContinue:(NSString *)nid withResult:(NetFunnelResult
     *)result
     {  // 대기가 발생되는 상황에서, TTL 주기마다 반복되어 호출된다.
             return NO;        // 대기창 표시: YES,(default)   대기창 미표시: NO
     }
```

- 대기가 발생되는 상황에서(background) 대기 제어를 종료하고자 하는 경우에 `netfunnel.StopContinue(); 호출을 통해 제어를 중지할 수 있습니다.

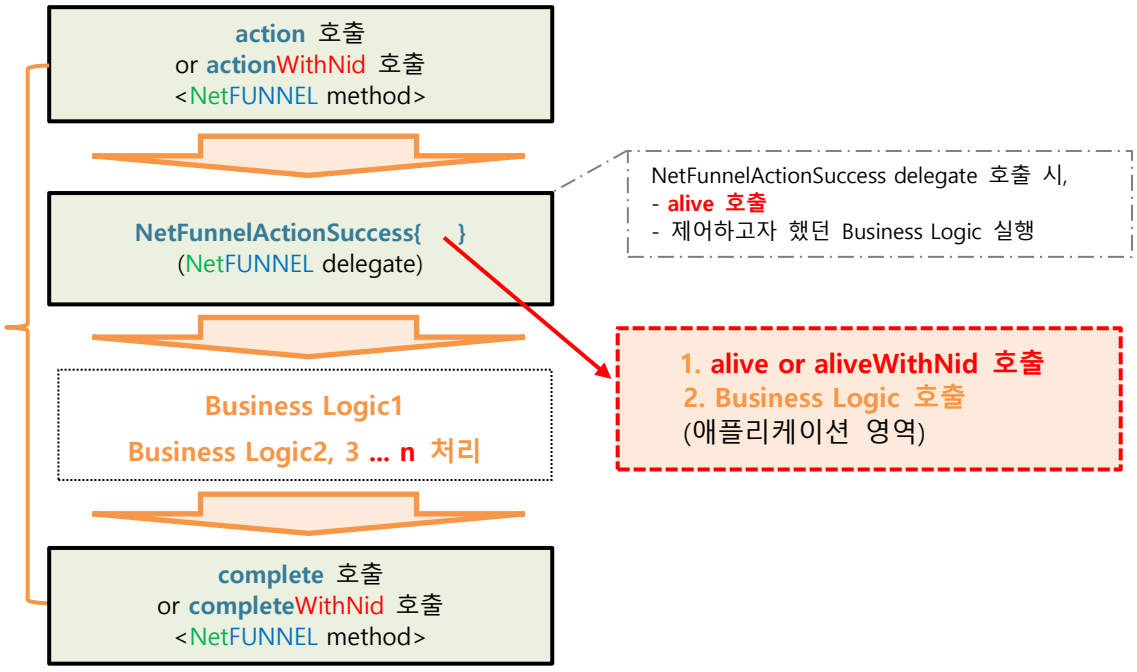
단일 구간 적용 방식

제어하고자 하는 Business Logic 을 넷퍼넬에서 제공하는 method(action, complete)로 감싸는 형태로 적용



여러 구간을 하나로 묶어서 적용하는 방식

제어하고자 하는 Business Logic 을 넷퍼넬에서 제공하는 method(action, complete)로 감싸고, NetFunnelActionSuccess delegate 호출 시점에 alive 를 호출하여 이후 여러 구간을 지나는 동안 timeout 에 빠지지 않게 구성하는 형태



㈜에스티씨랩

서울시 강남구 논현로 419 PMK빌딩 8층