블루투스를 활용한 틱택토 2 인대전 게임 개발

양원석
상명대학교 디지털미디어학과
e-mail : zmsenqn@naver.com

Development of Tic-tack-toe Game Application using Bluetooth for iPhones

WonSeok Yang
Dept. of Digital Media, Sang Myung University

요 약

요즘 다양한 스마트폰의 보급과 함께 스마트폰에서 활용되는 어플리케이션들이 주목 받고 있다. 그 중에 가장 활발히 어플리케이션이 개발되어 등록되어 있는 아이폰의 앱스토어는 개인 개발자들이 자발적으로 앱 개발에 참여할 수 있도록 유도하고 있다. 하지만 Xcode 라는 생소한 개발 툴을 이용하여 MacOS 라는 운영체제에서 개발을 하다 보니, 많은 기존 윈도우 기반의 스마트폰 개발자들이 아이폰 앱을 개발하는데 적지 않은 어려움이 있는 편이다. 본 논문에서는 아이폰에서 블루투스를 이용한 2 인용 대전 게임인 틱택토(Tick-tac-toe)를 C++기반으로 개발하여 그 내용을 소개한다.

1. 서론

최근 스마트폰의 보급과 함께 스마트폰을 활용한 다양한 앱(app)들이 개발되고 있다. 주목 받는 스마트폰 운영체제로는 iOS, 안드로이드, 윈도우 모바일 7 등이 있다. 그 중에서도 iOS 를 활용한 어플리케이션 개발이 가장 활발히 이루어지고 있다. 현재 아이폰의 앱 스토어에 등록된 앱의 개수는 25 만개 이상이며, 약 65 억 회의 다운로드를 기록하고 있다[1]. 이러한 앱들 중 페이스북과 구글어스가 가장 유명하다[2].

현재 국내의 많은 앱 개발자들이 아이폰 앱을 개발하려 하지만 생소한 Xcode 와 MAC 운영체제와 같은 친숙하지 못한 환경으로 인해 적지 않은 어려움을 겪고 있다. 그 중에서 큰 문제들 중하나는 코딩 스타일이 달라 그에 익숙해지는데 적지 않은 시간이 걸린다. 하지만 Xcode 는 supports C, C++, Fortran, Objective-C, Objective-C++, Java, AppleScript, Python, Ruby 등의 언어를 지원한다[3]. 본 논문에서는 C++를 기본으로 Xcode 의 gcc 컴파일러를 이용하여 기존의 코딩 스타일의 변경 없이 UI 부분을 제외한 핵심적인 코드는 C++로 아이폰 앱을 개발하여, 이를 소개하고자 한다.

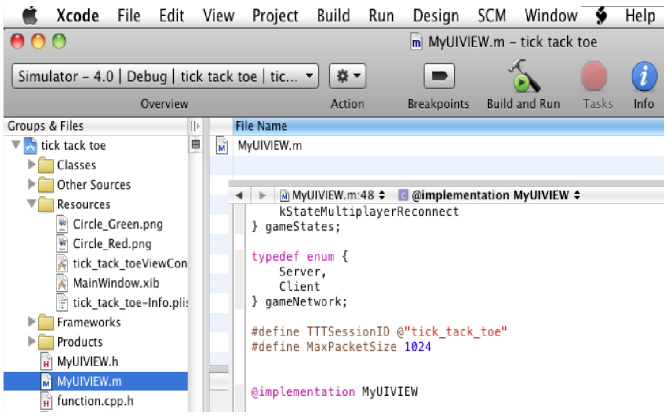
본 논문에서는 블루투스를 활용한 2 인용 대전 게임 틱택토(Tick-tac-toe)를 개발하였다. 틱택토는 오목과 유사한 게임이며, 오목과는 다르게 3x3 보드 상에서 진행되는데, 먼저 가로, 세로, 대각선으로 3 개의 말을 두는 사람이 이기는 게임이다.

본 논문은 아래와 같이 구성된다. 2 장에서는 아이폰 앱을 개발하기 위한 개발환경과 틱택토 게임에 대해서 간단히 소개하고, 3 장에서는 앱 개발에 사용된 함수들을 소개하며, 4 장에서는 게임의 실행과 진행을 설명하고, 마지막으로 5 장에서 결론을 맺는다.

2. Xcode 개발 환경과 틱택토

2.1. Xcode 개발 환경

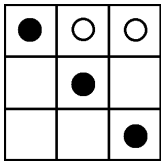
아이폰 앱을 개발하기 위해선 매킨토시 OS 가 탑재된 PC 에서 Xcode 를 사용해야 한다. Xcode 는 Objective-C 언어를 사용하며, 아이폰 시뮬레이터를 제공한다. [3]. 아이폰 개발을 위해서는 Apple 개발자 홈페이지 [4] 에서 개발자 인증 과정을 거쳐 Provisioning Certificate 를 획득하면, 개발한 앱을 아이폰에 직접 올려 실행 시켜 볼 수도 있다. 아이폰의 기본 개발환경은 다음 **[그림 1]**과 같다.



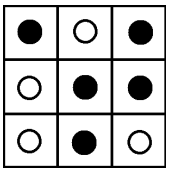
[그림 1] 아이폰 개발 환경

2.2. 틱택토 게임 규칙

틱택토는 오목과 유사한 게임이나, 오목과는 다르게 3x3 보드 상에서 진행되는데, 먼저 가로, 세로, 대각선으로 3 개의 말을 두는 사람이 이기는 게임이다. 보드에 9 개의 말을 다 두어도 승부가 나지 않은 경우는 무승부이다. 아래 **[그림 2]**와 **[그림 3]**은 틱택토의 승부가 난 경우와 비긴 경우를 각각 보여준다. 본 논문에서는 보드를 3x3 정수 배열로 표현하고 검정색 말을 1, 백색 말을 2 로, 그리고 비어있음을 0 으로 표현하였다.



[그림 2] 승부가 난 경우



[그림 3] 비긴 경우

틱택토는 간편하게 즐길 수 있는 인스턴트 게임으로 아이폰과 같은 모바일 환경에서 2 인이 대전용으로 하기에 매우 적합한 게임이다.

3. 틱택토 개발 코드

본 장에서는 틱택토를 위해 개발된 C++ 코드와 ObjectC 코드를 구분하여 활용하는 방법에 대해서 상세히 설명한다. 개발자가 직접 개발하는 주요 소스코드는 기존과 동일하게 C++로 작업할 수 있고, UI 와 블루투스와 같이 단말.기에 의존적인 소스는 ObjectC 를 이용한다.

3.1 틱택토 C++ 게임 코드

틱택토에 사용되는 주요 함수는 보드를 초기화 하는 함수와 게임 종료 여부를 판단하는 함수, 1 인 게임 모드 시 인공지능을 활용하는 부분으로 되어있으며, 각 함수는 C++로 제작되었다. 보드 초기화 함수는 게임 시작, 판정 직후에 작동하는데, 함수로 게임 보드 배열을 0 으로 초기화 시킨다. 그 코드는 [**그림 4]**와 같다

```
void initBRD(){
        memset(brd, 0, sizeof(int)*9); }
```

[그림 4] 보드 초기화 코드

승부 판정 코드는 게임의 종료 여부와 승패를 판정하는 코드이다. 2 장에서 설명한 게임 규칙을 기반으로 작성하였다. [**그림 5]**는 관련 코드이다.

```
int decision(){//승패를 확인
  //모든 승부를 확인,  0 은 비어있음
 if (board [0][0] && board [0][1] && board[0][2] )
return 1;
 …
 return 0;//승리 패턴 검색 실패}

 int decisionDraw() {
  int cnt =8;
  for(int i=0;i<9;i++)
  if(board[i/3][i%3]) cnt++;
  if (cnt==8) {return 1;//비김}
  return 0; //비기지 않음}
```

[그림 5] 승부 판정 코드

비기는 경우는 보드에 1 칸을 제외한 모든 칸이 차있는 경우이다. 마지막으로 인공지능 코드이다. 이 코드는 대전 모드가 아닌 1 인 플레이 모드로 실행했을 때 이용되며, 모든 상황에 대해서 컴퓨터가 유리한 조건을 찾아낸다. 게임의 재미를 더하기 위해서 랜덤으로 최적의 상황을 배제 하는 확률이 5%가 되도록 정하였다. 인공지능은 항상 백색 말을 이용한다.

```
void AI() {
        if (turn == true && brd[0][0] == 2 &&
brd[0][1] == 2 && brd[0][2] == 0) {
                brd[0][2] = 2; } // 인공지능
        else if (turn == true && brd[1][0] == 2 &&
brd[1][1] == 2 && brd[1][2] == 0) {
                brd[1][2] = 2; }
…
…
 if(random > 0.5){
      int* temp = FineFreeBoard();
      *temp = 2; }
}
```

[그림 6] 인공지능 코드

3.2 UI/블루투스 ObjectC 코드

아이폰 개발에 사용되는 코드 중 ObjectC 로 작성해야 하는 코드는 UI 관련 코드와 아이폰에서 제공하는 함수를 이용하는 코드이다. 주요 호출 함수만 이용을 하면 되고, 게임에 이용되는 모듈에 인자를 두어 값을 전달하거나 return 값으로 받아 ObjectC 코드를 최소화할 수 있다. 다음은 본 게임 개발에 사용된 주요 UI 코드와 터치 제어 코드, 그리고 블루투스 코드이다.

3.2.1. 화면 갱신 함수

[그림 7]의 코드는 게임을 위한 모든 시각적 요소들을 출력한다. 게임 보드의 경계선, 말, 점수, 차례를 화면에 출력한다. 또한 터치로 입력된 부분의 좌표값을 감지, 보정해 해당하는 칸에 말을 출력하는 기능을 한다.

```
 -(void)drawRect:(CGRect)Rect{
 //판을 그림
 CGContextRef VL1 =
UIGraphicsGetCurrentContext();
CGContextSetRGBFillColor(VL1, 0, 0, 0, 1);
CGRect rca = CGRectMake(106.7f, 0.0f, 1.0f, 320.0f);
 CGContextFillRect(VL1, rca);
 ...
 if(turn == true){
 for(UITouch *touch in self.touches){
  CGPoint pt = [touch locationInView:self];
 //현재 터치 위치에 대해 말의 배열 번호를 찾음
  gX = pt.x/107; gY = pt.y/107;
```

```
    if(brd[gY][gX] == 0){
        if(self.peerStatus == Server){
          brd[gY][gX] = 2; turn = false;
           [self sendNetworkPacket:gameSession
packetID:NETWORK_PLACE
   withData:brd ofLength:(sizeof(int)*9) reliable:NO];}
           else if(self.peerStatus == Client){
           brd[gY][gX] = 1; turn = false;
   [self sendNetworkPacket:gameSession
packetID:NETWORK_PLACE
   withData:brd ofLength:(sizeof(int)*9)
reliable:NO];}}}}}
   //보드를 화면에 그려줌
   for(int i=0;i<9;i++){
   if(brd[i/3][i%3] != 0){
    int ax = (i/3)*107+25.5;//각 말의 중심
    int ay = (i%3)*107+25.5;
   if(brd[i/3][i%3] == 1){
   CGRect r1 = CGRectMake(ax, ay, 60.0f, 60.0f);
   CGContextDrawImage(myContext,
r1,white.CGImage)
   if(brd[i/3][i%3] == 2){
   CGRect r2 = CGRectMake(ax, ay, 60.0f, 60.0f);
   CGContextDrawImage(myContext, r2,
black.CGImage)}}}
   //말을 둘 차례와 점수 출력
   …
   CGContextStrokePath(myContext);
}
```

[그림 7] 화면 출력 코드

3.2.2. 블루투스 제어 코드

블루투스 코드는 크게 4 부분으로 나뉜다. 첫번째는 피어를 검색하는 코드, 두번째는 패킷을 수신하는 코드, 세번째는 패킷을 송신하는 코드, 마지막으로는 연결상태를 체크하는 코드이다. 관련 코드는 **[그림 8]**에 있다.

```
//주변의 피어를 검색하여 연결한다
-(void)startPicker {
   …
   picker = [[GKPeerPickerController alloc] init];
   picker.delegate = self;
   [picker show]; }

//패킷 수신코드
- (void)receiveData:(NSData *)data fromPeer:(NSString
*)peer inSession:(GKSession *)session context:(void
*)context { …
   switch( packetID ) {
      //패킷 이벤트 처리
   }} …}

//패킷을 송신하는 코드
- (void)sendNetworkPacket:(GKSession *)session
packetID:(int)packetID withData:(void *)data
ofLength:(int)length reliable:(BOOL)howtosend {
  static unsigned char networkPacket[MaxPacketSize];
  const unsigned int packetHeaderSize = 2 * sizeof(int);
  if(length < (MaxPacketSize - packetHeaderSize)){
  int *pIntData = (int *)&networkPacket[0];
  // header info
    pIntData[0] = gamePacketNumber++;
pIntData[1] = packetID;

memcpy(&networkPacket[packetHeaderSize],data,length );
  NSData *packet = [NSData dataWithBytes:
networkPacket length: (length + packetHeaderSize)];
  //패킷을 전송
  [session sendData:packet
toPeers:[NSArray arrayWithObject:gamePeerId]
withDataMode:GKSendDataUnreliable error:nil];}}

//피어의 상태를 확인하는 코드
- (void)session:(GKSession *)session peer:(NSString
*)peerID didChangeState:(GKPeerConnectionState)state {
      if(self.gameState == kStatePicker)    {return; }
      if(state == GKPeerStateDisconnected) {
      if((self.gameState == StateMultiplayerReconnect) &&
self.connectionAlert && self.connectionAlert.visible) {
self.connectionAlert.message = message; } }
                        …
    self.gameState = kStateStartGame;    }
```

[그림 8] 블루투스 관련 코드

[그림 8]에서 이용된 함수들은 Game Kit 라이브러리[5]에 있는 함수이다. startPicker 함수는 블루투스 인터페이스로 게임할 상대를 검색하여 팝업으로 결과를 출력한다. receiveData 함수는 수신된 패킷을 분석하여 특정 패킷에 해당하는 동작을 수행한다. 게임이 실행중일 때 항상 동시에 실행되며 수신되는 패킷을 감시한다. 각각의 패킷의 구분을 위해서 switch 문에서 paketID 를 이용하여 각 패킷에 해당하는 동작을 구현하였다. switch 문에서는 packet ID 를 이용하여 각 패킷에 해당하는 동작을 구현하였다.

게임 모듈에서 각 상태에 따른 패킷 ID 와 게임 보드의 상태를 송신하기 위해 sendNetworkPacket 함수로 패킷을 송신하게 되는데, 게임 중 항상 작동되고 있는 수신 모듈과는 다르게 게임 모듈에서 상황에 알맞은 패킷 ID 와 정보를 송신할 때에만 작동한다.

마지막으로 session 함수는 피어의 연결 상태를 검사하여, 연결이 안된 경우에만 경고 팝업을 출력하고 초기 상태로 복귀한다. 상대 피어를 검색 중인 경우나 게임이 진행되고 있는 도중엔 별다른 동작을 취하지 않는다.

3.2.3. 터치 제어 코드

터치 제어 모듈은 터치 이벤트가 일어났을 때의 상황에 따른 동작을 결정한다. 게임이 시작되기 전인 상태라면 터치 시 피어 검색 상태로 동작하도록 하고, 게임 중이라면 터치된 부분에 말을 출력하고, 화면을 갱신한다. **[그림 9]**는 터치 제어코드를 보여준다.

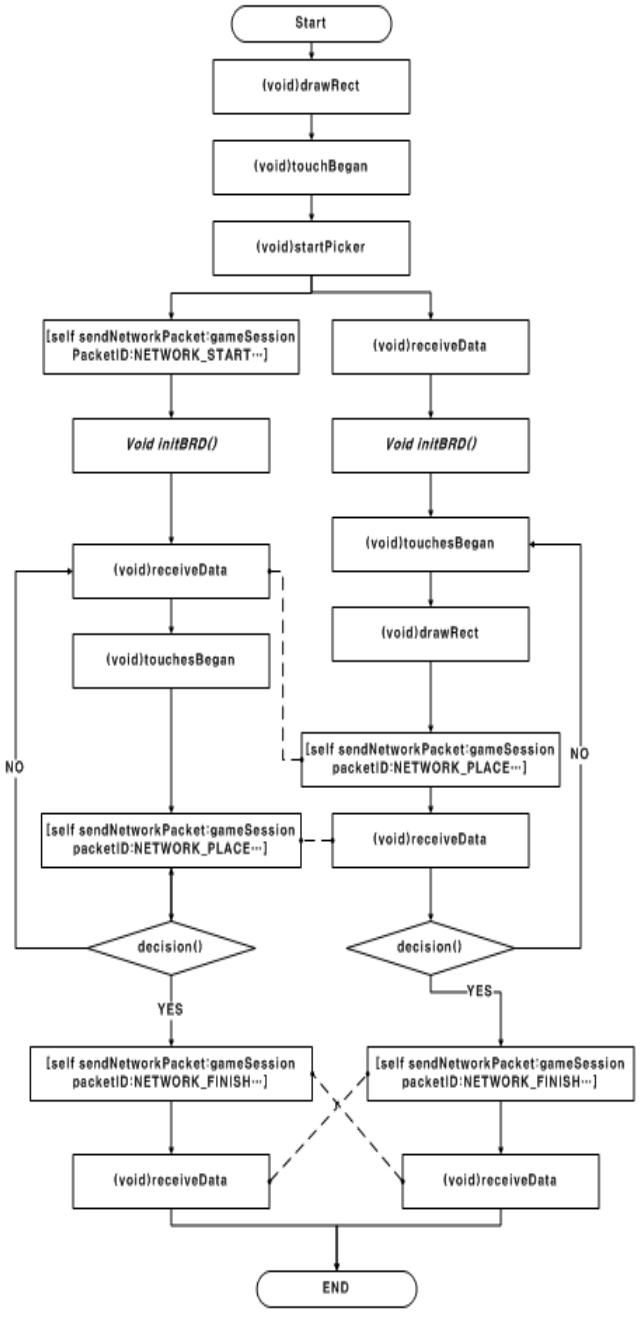
```
-(void)touchesBegan:(NSSet*)touches
withEvent:(UIEvent*)event {
            static bool gamestart = false;
```

```
if(!gamestart){
        [self startPicker];
        gamestart=true;        }
self.touches = [event allTouches];
[self setNeedsDisplay]; }
```

[그림 9] 터치 제어 코드

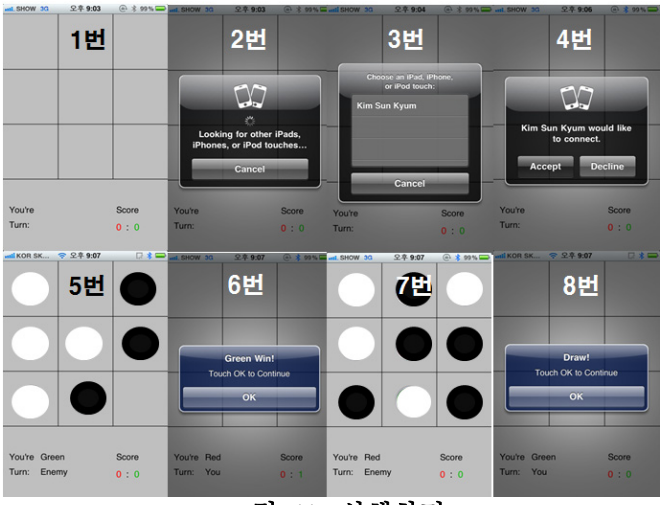
4. 게임 동작 루틴

[그림 10]은 3 장에서 소개된 함수들과 연관하여 틱택토 게임이 진행되는 흐름을 flow chart 형태로 나타내고 있다.startDraw()에서 분기는 플레이어 2 명의 각각 동작을 의미한다.



[그림 10] 함수 호출 플로우 차트

블루투스 모듈은 Xcode 내의 시뮬레이터로 테스트가 불가능하기 때문에 실제로 단말기에 올려서 테스트를 수행하였다. [그림 11]은 실제로 아이폰 상에서 실행시킨 화면을 보여주고있다.



[그림 11] 실행화면

[그림 11]의 1 번 그림은 게임 실행 시 초기화면이다. 2 번 그림은 주변의 블루투스로 사용자를 검색하는 화면이다. 3 번 그림은 주변의 사용자 리스트를 보여주는 화면이다. 4 번 그림은 데이터 커넥션을 만드는 화면이다. 5 번 그림은 게임이 백색 말이 승리한 경우를 보여주고, 6 번 그림은 '승리' 메시지를 표시한다. 7 번 그림은 비긴 경우이다. 마지막으로 8 번 그림은 '무승부' 메시지를 출력한 것이다.

5. 결론

본 논문에서는 틱택토 아이폰 앱을 개발하는 과정을 소개하였다. 앱 개발은 개발 도구인 Xcode 를 사용하여 C, C++ 언어로 코딩된 별도의 함수도 함께 호출하여 사용하였다.

개선될 점으로는 사용자의 플레이를 관전할 수 있는 모드와 훈수 모드 등을 추가하면 보다 여러 사람이 즐길 수 있는 틱택토 게임이 될 것이다.

참고문헌

[1] Apple Special Event, September 1, 2010.
[2]http://www.mobilewebgo.com/apples-most-popular-apps-ever-announced-april-2009.
[3] http://en.wikipedia.org/wiki/Xcode
[4] http://developer.apple.com
[5]http://developer.apple.com/library/ios/#documentation/NetworkingInternet/Conceptual/GameKit_Guide/Introduction/Introduction.html