DirectX ◆ Visual C++ ◆ 3ds max ◆ ZBrush

NR 25 (35) GRUDZIEŃ 2007

www.warp.pl

# WARP 2.0 Digital

numer 25

COMPUTER GRAPHICS - PROGRAMMING - GAME DEVELOPMENT

Aplikacje Windowsowe Microsoft Visual Studio 2005

3ds max
Modyfikacja HSDS

Wieści ze świata gier wideo:
- Activision przejmuje Bizarre Creations
- Bungie ponownie niezależne
- Rozdanie nagród BAFTA

*Najciekawsze informacje z największej w historii wystawy gier wideo w Japonii.*

TOKYOGAMESHOW2007

strona 08

# TOKYOGAMESHOW2007

**Wielka wystawa gier wideo w Japonii.**

**Witajcie,**

Witamy w kolejnym numerze WARP 2.0 Digital. W grudniowym wydaniu magazynu zamieściliśmy dla was kolejny artykuł z cyklu programowanie w środowisku Microsoft Visual Studio 2005. Tym razem dokładnie studiujemy temat tworzenia aplikacji pod Windows. W numerze znajdziecie także dział wiadomości ze świata gier wideo, w którym tym razem m.in. o zaskakujących manewrach Bungie.

Ponadto zapraszamy na wycieczkę do Tokyo, gdzie jak co roku odbyły się targi gier wideo. Dla grafików zamieszczamy kolejny artykuł opisujący pracę w 3ds max, tym razem opisujemy bardzo przydatną podczas modelowania modyfikację HSDS. A na końcu numeru galeria, w której występuje Artur Sadłos.

Życzymy miłej lektury!

*redakcja warp*


okładka **WARP 2.0 Digital 25**
Infinite Undiscovery
**© Square Enix**

## w numerze:

03 News - wieści ze świata gier wideo

08 Tokyo Game Show 2007 - wystawa gier wideo

14 Visual Studio 2005 - aplikacje Windowsowe

24 3ds max - modyfikacja HSDS

31 Galeria - Torturr

**WARP 2.0 Digital**
**Nr 25 (35) grudzień 2007**

**Zespół BCT**
Piotr Besta (matsuoka)
Daniel Besta (rabban)

**e-mail:**
blackcatwarp@o2.pl

**Internet:**
www.warp.pl



▲ Tokyo Game Show 2007 (panorama)

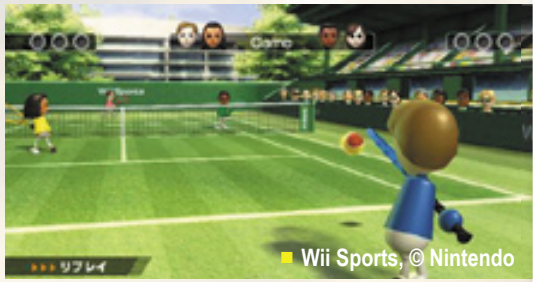
# 2007 British Academy Video Games Awards

## Nagrody BAFTA rozdane, sukces BioShock i Wii Sports.

We wtorek 23 października odbyło się piąte rozdanie nagród Brytyjskiej Akademii Sztuk Filmowych i Telewizyjnych (BAFTA) dla najlepszych gier wideo. Na ceremonii, która odbyła się w Londynie, triumfował opisywany przez nas w poprzednim numerze **BioShock** oraz gra **Wii Sports**. Wyprodukowana przez 2K Games

■ BioShock, © 2K Games

strzelanina BioShock zdobyła nagrodę w najbardziej prestiżowej kategorii Best Game. Natomiast tytuł Nintendo – Wii



■ Wii Sports, © Nintendo

Sports zagarnął aż sześć nagród spośród wszystkich trzynastu kategorii określanych przez akademię.

Cieszyli się również deweloperzy z Realtime Worlds, Capcom i Sony - gry **Crackdown, Okami** i **God of War II** zdobyły po dwie nagrody. Na nagrodę przyznawaną przez graczy głosowali w większości fani piłki nożnej i powędrowała ona do twórców **Football Manager 2007**. Warto również dodać, że **Will Wright**, twórca gier Sims i Sim City, otrzymał honorowe członkostwo akademii BAFTA. Jest to pierwszą postać ze świata gier wideo, która dostąpiła tego wyróżnienia. ■

BRITISH ACADEMY VIDEO GAMES AWARDS

**A oto pełna lista kategorii i zwycięzców tegorocznych nagród British Academy Video Game Awards:**

Action and Adventure:
**Crackdown**

Strategy and Simulation:
**Wii Sports**

Sports:
**Wii Sports**

Multiplayer:
**Wii Sports**

Casual:
**Wii Sports**

Use of Audio:
**Crackdown**

Original Score:
**Okami**

Technical Achievement:
**God of War II**

Artistic Achievement:
**Okami**

Gameplay:
**Wii Sports**

Innovation:
**Wii Sports**

Story and Character:
**God of War II**

Best Game:
**Bioshock**

The Gamers Award:
**Football Manager 2007 (PC)**

BAFTA's "Ones to Watch":
**Ragnarawk**

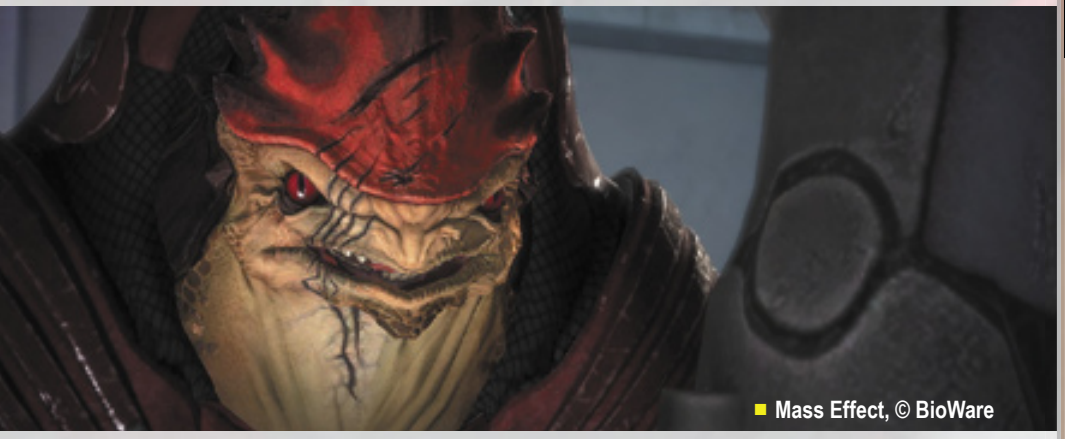
# Electronic Arts kupuje BioWare i Pandemic

## Wielkie zakupy amerykańskiego giganta.

11 października ogłoszono zawarcie umowy pomiędzy Electronic Arts a **Elevation Partners**. Dotyczy ona zakupu **VG Holding Corp.** - firmy która została uformowana w 2005 roku w celu połączenia dwóch innych firm – kanadyjskiego producenta gier wideo **BioWare** oraz kalifornijskie **Pandemic Studios**.

Teraz obie firmy trafiają w ręce Electronic Arts, a wartość całej transakcji wyniesie 860 milionów dolarów. Przejmowani deweloperzy zatrudniają obecnie około 800 osób, które pracują w studiach rozmieszczonych w Kanadzie, Stanach Zjednoczonych i Australii. W wyniku transakcji Amerykański wydawca przejmuje również prawa do niektórych produkowanych przez te firmy tytułów, są to m.in. futurystyczny **Mass Effect**, **Jade Empire** i **Mercenaries**. Całkowite przejęcie BioWare i Pandemic nastąpi w styczniu 2008.

Firma BioWare znana jest z takich gier jak chociażby **Baldur's Gate**, czy też niezwykle popularny tytuł role-playing **Star Wars: Knight of the Old Republic**. Z kolei w studiu Pandemic powstały takie gry jak **Mercenaries** i **Full Spectrum Warrior**. ■



■ Mass Effect, © BioWare

# 3ds Max 2008 trial

## Testowa wersja nowej odsłony popularnego pakietu 3D.

Firma Autodesk udostępniła na swojej stronie darmową wersję trial programu 3ds Max 2008. Można ją pobrać z oficjalnej strony firmy, link do wymaganego formularza znajduje się tutaj.

Po wypełnieniu formularza możemy przystąpić do ściągania pliku, który zajmuje 568 MB. Program można uruchomić na Windows Vista i Windows XP, ponadto 3ds Max 2008 obsługuje DirectX 10. ■

# Autodesk kupił Skymatter

## Mudbox w rękach producenta pakietów 3ds Max i Maya.

Mieszczące się w Nowej Zelandii studio **Skymatter Limited** zostało przejęte przez giganta wśród oprogramowania 3D, firmę Autodesk.

Firma Skymatter jest producentem programu **Mudbox**, który jest przeznaczony dla artystów i designerów. W ostatnim czasie stał się on niezwykle popularny w przemyśle filmowym oraz gier wideo. Wykorzystywany był m.in. podczas produkcji filmu King Kong.

Mudbox posiada innowacyjny zestaw narzędzi, który umożliwia rzeźbienie z wykorzystaniem pędzla. Program został zaprojektowany przez profesjonalnych grafików, dzięki czemu posiada bardzo przyjazny interfejs. Najczęściej jest on wykorzystywany do tworzenia złożonych, realistycznych modeli 3D. Dzięki tej transakcji możliwości programu mogą zostać jeszcze bardziej rozbudowane, dzięki współpracy z niezwykle doświadczonymi zespołami wchodzącymi w skład Autodesk. ■

Project Gotham Racing 4 (Xbox 360), © Microsoft

*„Bizarre Creations jest wiodącym niezależnym deweloperem z doświadczeniem w tworzeniu udanych i oryginalnych produktów, zwłaszcza na polu gier wyścigowych. Studio to należy do jednych z najbardziej kreatywnych i innowacyjnych w całym przemyśle i jesteśmy niezwykle podekscytowani tym, że możemy przywitać ten utalentowany zespół w Activision."*

**Robert Kotick, CEO firmy Activision**

# ACTIVISION przejmuje Bizarre Creations

## Twórcy serii Project Gotham Racing w rękach amerykańskiego giganta.

Firma Activision przeprowadziła w ostatnim czasie strategiczną operację, która ma jej umożliwić włączenie się do rywalizacji w kategorii gier wyścigowych. Na celowniku znaleźli się autorzy niezwykle popularnej serii wyścigów Project Gotham Racing. Do tej pory, w Stanach Zjednoczonych i Europie sprzedało się ponad 4,5 miliona egzemplarzy gier z tej serii (jest to wynik bez czwartej odsłony, której premiera miała miejsce 2 października 2007).

Pod koniec września brytyjska firma Bizarre Creations zamieściła na swojej oficjalnej stronie informację o przejęciu przez Activision. Od tej pory Bizarre funkcjonować będzie pod skrzydłami tego wielkiego wydawcy i przestaje być deweloperem

**Wartość sektora gier wyścigowych stanowi obecnie aż 10% całego światowego rynku gier wideo. Zatem nic dziwnego, że Activision tak bardzo zależy na posiadaniu własnych udziałów w tej części przemysłu. Z prawej screen z najnowszej gry Bizarre, Project Gotham Racing 4.**

całkowicie niezależnym. Przedstawiciele firmy oświadczyli, że transakcja nie jest spowodowana jakimiś kłopotami finansowymi, a jest to jedynie kolejny krok w rozwoju tej grupy. Przyłączenie do Activision nie pociąga za sobą żadnych zmian w wizerunku, firma Bizarre Creations w dalszym ciągu funkcjonować będzie pod swoją dotychczasową nazwą.

Firma Bizarre Creations znana jest głównie za sprawą serii gier Project Gotham Racing. Dwie pierwsze gry z tej serii ukazały się na konsolę Xbox, natomiast trzecia i czwarta na Xbox 360. Bizarre posiada obecnie dwa zespoły, które pracują nad innymi grami. Pierwszy z nich (Amax team) jeszcze niedawno pracował nad Project Gotham Racing 4 (wydawca Microsoft), natomiast drugi zespół (Shitstorm team) pracuje obecnie nad zapowiedzianą na rok 2008 grą The Club (wydawca SEGA).

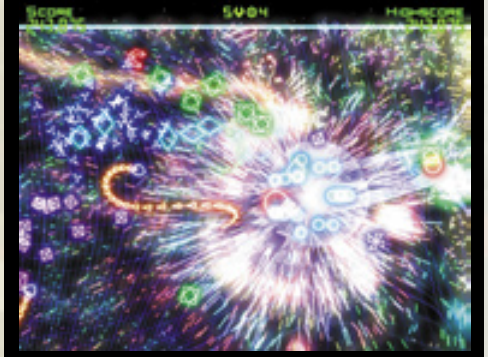
W najbliższej przyszłości dwa zespoły Bizarre rozpoczną pracę nad dwoma zupełnie nowymi tytułami. Jednym z nich mają być wyścigi, które nie będą w żaden sposób powiązane z Project Gotham Racing (prawa do tej marki posiada Microsoft). Będzie to zupełnie nowy projekt, przeznaczony na różne platformy. Natomiast drugi zespół również rozpocznie prace nad nowym projektem, o tej grze wiadomo tylko tyle, że nie będą to wyścigi i pojawią się w niej postacie (character game). Pierwszy tytuł Bizarre Creations dla Activision ukaże się najwcześniej w 2009 roku. ■

▲ Nowa produkcja Bizarre Creations zatytułowana The Club powstaje dla wydawcy SEGA , a jej premiera zapowiedziana jest na początek roku 2008. Jest to gra akcji third-person z arcadową rozgrywką, która polega na zdobywaniu jak największej liczby punktów.

***„Celem Bizarre Creations zawsze było dostarczanie naszym kreatywnym zespołom nowych, ciekawych wyzwań. Funkcjonowanie jako niezależne studio Activision nie odbiera nam wpływu na podejmowanie decyzji podczas tworzenia nowych ekscytujących produktów. Finansowe i marketingowe wsparcie Activision pozwoli nam rozwijać się w oparciu o nasz aktualny sukces komercyjny i wyniesie naszą firmę oraz nasze gry na jeszcze wyższy poziom.”***

**Martyn Chudley, managing director firmy Bizarre Creations**

**Firma Bizarre Creations założona została w 1988 roku przez Martyn'a Chudley'a (do 1994 roku funkcjonowała pod nazwą Raising Hell Software). Na początku swego istnienia zespół liczący jedynie 5 osób rozpoczął pracę nad nowym projektem. Po pewnym czasie powstała z tego wersja demo, którą zaprezentowano firmie Psygnosis. Wynikiem tej prezentacji było podpisanie umowy na stworzenie wyścigów Formula 1 na konsolę PlayStation. Zespół rozrósł się wówczas do 11 osób, a gra powstała w przeciągu 14 miesięcy. Tytuł ten odniósł później spory sukces – był najlepiej sprzedającą się grą w Europie w 1996 roku. Kolejną grą na koncie były wyścigi Metropolis Street Racer na konsolę Dreamcast. Po tym projekcie firma przystąpiła do prac nad Project Gotham Racing. Wyścigi ukazały się w 2001 roku jako tytuł startowy dla konsoli Xbox i były drugą najlepiej sprzedającą się grą z wszystkich tytułów, które ukazały się na starcie tej platformy... Studio Bizarre mieści się w Liverpoolu.**



▲ Bizarre Creations ma na swoim koncie również popularną grę Geometry Wars, która ukazała się na Xbox 360 (Xbox Live Arcade) i platformę PC.

**Pierwszy niezależny producent i wydawca gier wideo. Activision jest obecnie jednym z największych dystrybutorów gier wideo na świecie. Firma została założona 1 października 1979 roku przez pracownika przemysłu muzycznego Jim'a Levy i czterech byłych programistów Atari - David'a Crane'a, Larry'ego Kaplan'a, Alan'a Miller'a i Bob'a Whitehead'a. Pierwszym produktem Activision były cartridge do konsoli Atari 2600. Na przestrzeni lat firma wydała takie tytuły jak Pitfall!, Interstate 76', Civilization: Call to Power, Tony Hawk's Pro Skater, czy też Quake III Arena i Doom 3. Obecnie w skład Activision wchodzi ponad dziesięć zespołów deweloperskich w Stanach Zjednoczonych i Kanadzie, a zaliczają się do nich m.in. Raven Software (Black Crypt, Hexen, Heretic, Quake 4) i Infinity Ward (seria Call of Duty).**

*„Praca z Microsoft układała się świetnie, pozwoliło nam to rozwinąć się jako zespół i tworzyć ambitne, przebojowe gry, nad którymi wszyscy chcieli pracować. I w dalszym ciagu będą dla nas świetnym partnerem. Bungie jest jak rekin, aby żyć musimy być w stałym ruchu. Musimy bez przerwy się sprawdzać, albo możemy być delfinami, lub krowami morskimi..."*

**Jason Jones, jeden z założycieli firmy Bungie**

# Bungie ponownie niezależne

## Microsoft zachowuje prawa do serii Halo i pozostaje wydawcą gier Bungie.

Zaraz po premierze Halo 3, która miała miejsce 25 września, pojawiły się plotki o rozłamie pomiędzy Bungie i Microsoft. Wieści na ten temat rozprzestrzeniały się w sieci w błyskawicznym tempie, ale dla większości były zupełnie nieprawdopodobne. Już na początku października wszystko się jednak wyjaśniło, świat obiegła bowiem wiadomość o tym, że twórcy serii Halo ponownie staną się całkowicie niezależną firmą. Gigant z Redmond pozostanie wydawcą Bungie i zachowa prawa do serii Halo.

Bungie zostało przejęte w 2000 roku przez Microsoft, a celem tej transakcji była produkcja gier na pierwszą konsolę Xbox. Od tamtej pory studio wypuściło trzy części strzelaniny sci-fi Halo. Pierwsze dwie odsłony ukazały się na konsolę Xbox, a trzecia na Xbox 360. Każda gra z tej serii odniosła ogromny sukces komercyjny, a Halo 3 w ciągu zaledwie pierwszego tygodnia sprzedało się w liczbie bliskiej 3,3 miliona egzemplarzy i zarobiło ponad 225 milionów dolarów. Do tej pory trzy odsłony serii Halo sprzedały się łącznie w liczbie ponad 16,5 miliona kopii, a z pewnością w najbliższym czasie Halo 3 jeszcze swój wynik poprawi.

Przedstawiciele Bungie określają nową sytuację ekscytującą ewolucją stosunków pomiędzy partnerami. Zapewniają, że w dalszym ciągu będą współpracować z firmą Microsoft nad serią Halo, dodatkami do gry oraz ewentualnymi kolejnymi projektami. Wygląda więc na to, że w najbliższej przyszłości nie zauważymy wielkich zmian w działalności Bungie. Jak będzie w przyszłości, czas pokaże, ale pewne jest że o Bungie jeszcze usłyszymy. ■


▲ Halo 3 podbiło niemal wszystkie listy najlepiej sprzedających się gier wideo. W swoim pierwszym tygodniu po premierze gra znalazła się nawet na pierwszym miejscu listy w Japonii, a przecież nie jest tajemnicą że konsola Microsoft'u akurat w kraju kwitnącej wiśni nie radzi sobie najlepiej. W ciągu ostatnich kilku dni października w Japonii rozeszło się blisko 60 000 egzemplarzy trzeciej części przygód Master Chief'a.

# TOKYO GAME SHOW 2007

## LINK UP, REACH OUT, TO THE WORLD.

20 - 23 WRZESIEŃ

**Najciekawsze informacje z największej w historii wystawy gier wideo w Japonii.**

DANIEL BESTA A.K.A. RABBAN

W dniach 20 – 23 września, w **Makuhari Messe Convention Center** w Japonii odbyła się wystawa Tokyo Game Show 2007. Wydarzenie to organizowane jest przez **Computer Entertainment Supplier's Association** już od 1996 roku i skupia wszystkich zainteresowanych przemysłem gier wideo, zarówno producentów gier, sprzętu, jak i samych graczy. Tegoroczna impreza była rekordowa jeżeli chodzi o liczbę wystawców, przestrzeń przeznaczoną na stoiska oraz liczbę uczestników.

Na TGS2007 zaprezentowało się łącznie 217 firm, fundacji i szkół z 19 krajów, a liczba zaprezentowanych gier przekroczyła 700 tytułów. Targi zwiedziło wraz z przedstawicielami mediów aż 193 tysiące widzów. Z tego też powodu, mimo dużej powierzchni centrum Makuhari Messe, w godzinach napływu największej fali zwiedzających niektóre

▲ Panorama ukazująca stoiska w jednej z hal centrum Makuhari Messe.

przejścia pomiędzy stoiskami były tak zatłoczone, że aż trudno było się w nich poruszać.

Po raz pierwszy w historii targi odbywały się przez cztery dni – o jeden dzień dłużej niż w poprzednich latach. Pierwsze dwa dni określone były jako Business Days i przeznaczone były jedynie dla przedstawicieli mediów i zaproszonych gości. Dla szerszej publiczności otwarte były jedynie ostatnie dwa dni imprezy.

## Prezentacja Sony

Spośród wszystkich firm biorących udział w imprezie, największym stoiskiem pochwalić się mogło Sony. Zaraz po ceremonii otwarcia TGS, na scenie w wypchanej po brzegi sali konferencyjnej Makuhari Messe pojawił się **Kaz Hirai**. Prezes Sony Computer Entertainment wygłosił przemówienie zatytułowane **„A Look at Expanding Our Business Strategy Toward New Growth: The Expanding PlayStation World”**, w którym przedstawił najbliższe plany firmy.

Na początku swojego wystąpienia Kaz Hirai wspomniał co nieco o dotychczasowych sukcesach Sony - konsola PlayStation 2 sprzedała się w liczbie 120 milionów egzemplarzy, a PSP zbliża się właśnie do osiągnięcia wyniku 30 milionów. Kilka chwil poświęconych zostało również nowym możliwościom konsolki PSP.

Na wybrane rynki Sony wprowadza dodatkowe gadżety o które zabiegali klienci, jest to m.in. dołączana kamera (Europa i Japonia) oraz 1SEG TV Tuner (tylko Japonia), który umożliwia oglądanie telewizji na ekranie PSP. Zademonstrowano nowe możliwości współpracy pomiędzy PSP a konsolą PS3 (PSP możemy wykorzystać np. jako dodatkowy ekran, czy też kontroler).


**Screen i postacie z Devil May Cry 4, © Capcom. Gra zapowiedziana jest już na początek 2008 roku (w Europie premiera przewidziana jest na luty), a ukaże się na platformy PC, PlayStation3 i Xbox 360. Seria Devil May Cry cieszy się dużą popularnością, zwłaszcza wśród hardcorowych graczy (ze względu na wysoki poziom trudności). Trzy pierwsze części sprzedały się na całym świecie w liczbie ponad 5 milionów egzemplarzy. Czwarta odsłona jest pierwszym tytułem na nową generację konsol, w grze pojawi się nowy główny bohater Nero.**

Oprócz kilku nowych rozwiązań, w trakcie tego wystąpienia zaprezentowano serię ujęć z różnych gier, które właśnie się ukazały lub pojawią się już niebawem na PlayStation 3. Pokazano co nieco z takich tytułów jak Devil May Cry 4,

Heavenly Sword, Call of Duty 4, a także Metal Gear Solid 4 i Afrika. Następnie padło kilka słów na temat wprowadzania nowych rozwiązań, które pozwolą na poprawę współpracy z zaprzyjaźnionymi producentami. Priorytetem dla Sony w najbliższej przyszłości ma być budowa lepszych stosunków z bliskimi partnerami oraz udoskonalenie środowiska dla deweloperów.

Zgodnie z wcześniejszymi przewidywaniami zaprezentowany został również **Dual Shock 3**, nowy pad do PlayStation 3. Dzięki niemu również na najnowszej konsoli Sony będziemy mogli odczuwać wibracje w trakcie gry. Pad wygląda identycznie jak Sixaxis, ale w środku posiada wbudowany system rumble. Początkowo Sony twierdziło, że jest to technologia poprzedniej generacji i czas wibracji już minął. Zamiast kolejnej wersji Dual Shock powstał Sixaxis, który zamiast wibracji posiadał czujnik ruchu – motion sensitive. Szybko jednak okazało się, że był to duży błąd gdyż gracze na całym świecie zaczęli domagać się powrotu trzęsącego się w rękach pada.


■ **Kaz Hirai prezentuje Dual Shock 3. Pad pojawi się w sprzedaży najpierw w Japonii (listopad 2007), a dopiero później w Europie i Stanach Zjednoczonych (początek 2008).**

# Evolution w rękach Sony

## Gigant kupuje studio w którym powstaje MotorStorm 2

W trakcie głównej prezentacji Sony na Tokyo Game Show 2007, **Kaz Hirai** ogłosił przejęcie europejskiej firmy **Evolution** wraz z ich drugim studiem **Bigbig**. Studio Evolution jest odpowiedzialne za wielki hit na PS3, wyścigową grę MotorStorm.

Firma Evolution mieści się w Runcorn, niewielkiej miejscowości w pobliżu Liverpoolu. Założona została w 1999 roku przez Martin'a Kenwright'a i Ian'a Hetherington'a. Przez lata grupa współpracowała z Sony tworząc serię gier rajdowych **WRC** na PS2. Dodatkowe studio Bigbig powstało w 2001 roku i pracowało nad serią gier **Pursuit Force** (platformy PS2 i PSP). Obecnie główne studio pracuje nad dodatkami do MotorStorm oraz nad drugą odsłoną tych wyścigów.

Natomiast w Bigbig Studios powstaje obecnie gra Pursuit Force: Extreme Justice, która zawita na kosnolkę PSP na początku 2008.

Cała inwestycja kosztowała Sony 16 milionów funtów. Na transakcji najbardziej wzbogacił się Martin Kenwright, który posiadał 75% udziałów Evolution - na jego konto trafiło 12 milionów funtów. Większość z pozostałej części udziałów była własnością Hetherington'a. Obydwu panów nie masz już w Evolution, a firma należy teraz do wielkiej grupy SCE Worldwide Studios. Szefem zespołów mieszczących się w Liverpoolu został Mick Hocking, który wcześniej obejmował stanowisko Managing Director w Evolution. ■

▲ Metal Gear Solid 4, © Konami. Na imprezie w Tokyo po raz pierwszy udostępniono zwiedzającym grywalne demo tej świetnie zapowiadającej się gry.

Na reakcję Sony nie trzeba było długo czekać, firma zawarła kosztowną umowę z Immersion Corporation, która to posiadała prawa do technologii force-feedback wykorzystanej w padzie Dual Shock 2. Trzecia wersja Dual Shock będzie zatem posiadała zarówno motion sensitivity jak i rumble. Już teraz wiadomo, że co najmniej 60 tytułów na PS3 będzie korzystać z nowej/starej funkcji (wśród nich jest m.in. Metal Gear Solid 4). Natomiast złą informacją ujawnioną na prezentacji było przesunięcie premiery serwisu **PS3 Home** na wiosnę 2008. Pierwotnie Home miał być dostępny już pod koniec tego roku.

## MGS4, Arfika i Flower

Spośród nadchodzących gier na PS3 chyba największą uwagę przykuwała kolejna część przygód Solid Snake'a. Czwarta odsłona Metal Gear Solid ma się ukazać już na początku przyszłego roku. Firma Konami przygotowała na wystawę specjalne stoisko o wyglądzie bazy wojskowej, gdzie można było po raz pierwszy zagrać w nowy tytuł Hideo Kojimy. Oprócz

■ Afrika, © SCE

gry **Metal Gear Solid 4: Guns of the Patriots** prezentowano tutaj również **Metal Gear Solid Online** i kolejną odsłonę piłkarskiej serii Pro Evo.

Innym ciekawie zapowiadającym się tytułem na PlayStation 3 jest symulacja afrykańskiej dziczy o roboczym tytule **Afrika**. Ze zwiastunów wynika, że rozgrywka pozwoli na obserwowanie m.in. ryczących lwów, pasących się antylop, grubych hipopotamów i żyraf. Na uwagę zasługuje zwłaszcza bardzo realistyczna animacja wszystkich zwierząt.

Jedną z niewielu całkowicie nowych zapowiedzi na tegorocznych targach był tytuł **Flower**, który jest kontynuacją popularnej gry flOw (jest ona dostępna w sieciowym serwisie PlayStation Network). Obecnie nie wiadomo jeszcze zbyt wiele na temat nowej gry firmy thatgamecompany. Akcja rozgrywać ma się na polu kwiatów, a gameplay najprawdopodobniej będzie podobny do gry flOw. Natomiast na konsolkę PSP zapowiedziano tytuł Secret Agent Clank, gdzie wcielimy się w postać Clank'a z serii Ratchet & Clank.

## Nowe japońskie gry RPG na X360

W kraju kwitnącej wiśni konsola Xbox 360 nie radzi sobie zbyt dobrze. Mimo, że od premiery tego systemu minęło już 2 lata, to wyniki sprzedaży w Japonii nie wyglądają najlepiej. Z informacji podanych przez Microsoft wynika, że na całym świecie do lipca 2007 sprzedało się 11,5 miliona konsol. Większość z tej liczby, bo aż 10 milionów konsol, rozeszło się jednak w Europie i USA. W Japonii liczba nabywców to zaledwie około 500 tysięcy osób.

# The Japan Game Awards 2007

## Nintendo i Capcom na najwyższym stopniu podium.

W trakcie Tokyo Game Show odbyło się rozdanie nagród The Japan Game Awards 2007, które przyznaje organizacja **Japan's Computer Entertainment Software Association**. W konkursie pod uwagę brane były gry wydane w okresie od 01.04.2006 do 31.03.2007. Główna nagroda Game of the Year trafiła egzekwio do dwóch tytułów.

Jednym ze zwycięzców okazała się gra **Wii Sports**. Produkcja Nintendo miała bardzo duży wpływ na sukces konsoli Wii. Na całym świecie sprzedała się bowiem w liczbie ponad 8 milionów egzemplarzy. W osiągnięciu tego wyniku z pewnością pomógł jej fakt, że wszędzie poza Japonią znajdowała się w zestawie z konsolą. Uzasadnieniem wyboru Wii Sports był wielki wkład w zmianę wizerunku konsolowej rozrywki i to, że twórcom udało się dotrzeć do całej masy nowych odbiorców w różnym wieku.


Drugim zwycięzcą okazał się tytuł **Monster Hunter Portable 2nd** (poza Japonią gra nosi tytuł Monster Hunter Freedom 2). Jest to niezwykle popularna gra w Japonii, która stałą się tam czymś więcej niż tylko grą wideo. Gracze, przede wszystkim w wieku szkolnym, wykorzystują ja bowiem również jako narzędzie do komunikacji. W samej Japonii gra sprzedała się w liczbie przekraczającej milion egzemplarzy. Producentem MHP2 jest firma Capcom.

Na uroczystości JGA rozdano również nagrody w innych kategoriach. Nagrodą Best Sales Award wyróżniono najlepiej


sprzedający się tytuł. W tej kategorii triumfowała produkcja Nintendo – **Pokemon Diamond/Pearl** (platforma Nintendo DS). W ciągu roku rozeszło się blisko 5 milionów kopii tej gry. Nagroda Global Award Japanese Product powędrowała do twórców z Capcom za **Dead Rising**, a Global Award Foreign Product otrzymała firma Epic Games za strzelaninę **Gears of War**. Kilkanaście gier zostało wyróżnionych za jakość, wśród nich znalazły się m.in. Lost Planet, Okami i Blue Dragon ■

Mimo słabych wyników sprzedaży, Microsoft się nie poddaje i po raz kolejny podejmuje walkę o ten rynek. Po raz kolejny pomóc ma w tym **Hironobu Sakaguchi**, twórca serii Final Fantasy. Jeden z największych twórców RPG poprzednio pracował nad grą Blue

▲ **Screeny z Lost Odyssey, premiera gry już w lutym 2008. Nad tym tytułem pracuje Mistwalker - studio założone przez Hironobu Sakaguchi'ego oraz innych byłych pracowników Square Enix.**

Dragon, która również ukazała się wyłącznie na Xbox 360. Obecnie natomiast trwają ostatnie prace nad zaprojektowaną przez niego grą **Lost Odyssey**. Oprócz tej produkcji, na targach zaprezentowano również **Infinite Undiscovery** - inny RPG, który powstaje w studiu Square Enix z myślą o rynku

japońskim i podobnie jak Lost Odyssey ukaże się tylko na konsolę Microsoftu. Na zwiastunach obydwie gry prezentuję się świetnie, ale musimy jeszcze poczekać z odpowiedzią na pytanie czy poprawią one sytuację tej platformy w Japonii.

Bardzo duże zainteresowanie wzbudziła powstająca w studiu Team Ninja gra

▲ Infinite Undiscovery, © Square Enix

**Ninja Gaiden 2** (również exclusive na X360). Ta niezwykle brutalna gra ukaże się w 2008 roku i już teraz zapowiada się na spory hit. Kilka ciekawych pozycji zapowiedziano na Xbox Live Arcade. Już wkrótce pojawi się tutaj nieco podrasowana wersja kultowej gry **REZ**, bardzo oryginalna strzelanka, w której dużą rolę odgrywa muzyka. Oprócz tego na XBLA trafią gry **Every Extend Extra Extreme** i **Ikaruga**.

## Nintendo odpoczywa

Obecnie w Japonii króluje Nintendo dzięki sukcesom NintendoDS i Wii. W tym roku jednak zabrakło tej firmy na targach. Mimo to na TGS2007 można było zobaczyć i przetestować sporo gier na te dwie konsole. Duże zainteresowanie wzbudzał tytuł **Resident Evil: Umbrella Chronicles** firmy Capcom, który prezentowany był w stoisku wystylizowanym na nawiedzony

▲ Ninja Gaiden II, © Tecmo

dom. Wśród zapowiedzi na Nintendo DS ciekawostką jest gra **Populous DS**, remake hitu Peter'a Molyneux'a z 1989 roku. Nad tą produkcją czuwa zespół EA Japan, który jest odpowiedzialny za grę SimCity DS.

Wymienione powyżej tytuły to jedynie garstka z tego co zaprezentowano na targach przez cztery dni. Tegoroczne Tokyo Game Show uznane zostało za niezwykle udane, a następna wystawa w 2008 roku odbędzie w październiku. ■

**Na wstępie witam wszystkich początkujących programistów DirectX i OpenGL. W ostatnich miesiącach dostaliśmy od Was kilkanaście e-maili z prośbą o ponowne dokładne przedstawienie procesu budowy i działania aplikacji windowsowej. Dziwi nas to, bowiem ten temat był wałkowany już dwa razy w poprzednich numerach. Widocznie nie dość dokładnie. Dlatego niniejszy artykuł wyczerpuje ten temat dogłębnie. Został przygotowany niemal od zera i w najdrobniejszym szczególe omawia proces tworzenia aplikacji dla Windows. Ze względu na rozmiar podzieliliśmy go na dwie części. Zaznaczamy, że jest to ostatni artykuł na ten temat jaki został opublikowany w WARP Digital 2.0. Zastrzegamy, że nawet dla następnej wersji środowiska VS nie będziemy tworzyć podobnego artykułu :)**

Microsoft Visual Studio 2005

# Aplikacje Windowsowe

*Piotr Besta a.k.a. matsuoka*

Nauka programowania aplikacji dla Windows to tak naprawdę nauka Windows API, którą rozpoczniemy od zbudowania niewielkiej, bardzo prostej, okienkowej aplikacji. Na początku pokażę jak utworzyć projekt takiej aplikacji, jak go skompilować i ostatecznie uruchomić. Oprócz tego będziemy mówić o wszystkim co jest związane z funkcjonowaniem budowanego programu. Omówimy bardzo dokładnie kilkanaście podstawowych funkcji Windows API. Od najprostszej jak MessageBox wyświetlającej okienko komunikatu po dużo bardziej zaawansowane. Nauczysz się odbierać wiadomości okna, które Windows wysyła do aplikacji. Dowiesz się jak napisać tzw. procedurę okna, która zajmuje się ich przetwarzaniem. Nauczysz się tworzyć, wyświetlać i poprawnie zamykać okno programu. Powiemy także sporo o pracy w Visual Studio 2005.

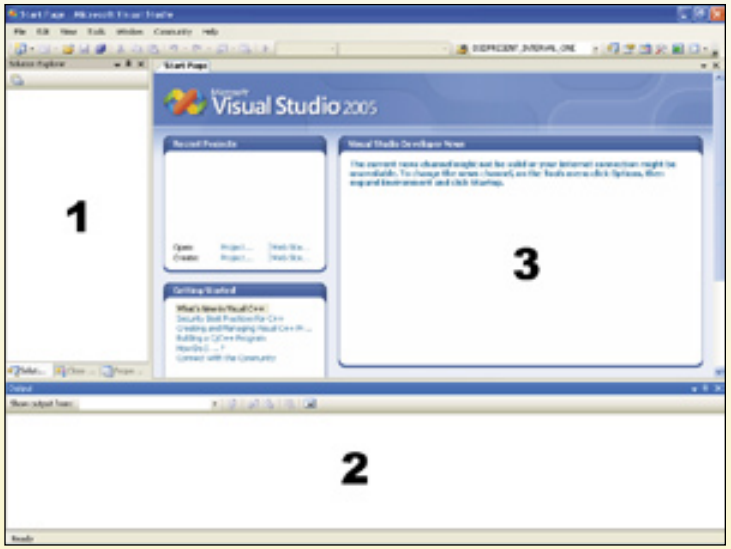
## Pierwszy program

Zgodnie z tym co napisałem powyżej, najpierw nauczymy się tworzyć prostą aplikację dla systemu Windows. Ten artykuł pozwoli ci zapoznać się w minimalnym stopniu ze środowiskiem Visual Studio 2005 (8.0). Zatem nie ma na co czekać, uruchom środowisko. Zapnij pasy bezpieczeństwa. Startujemy!

Po uruchomieniu środowiska programistycznego Microsoft Visual Studio 2005, ekran Twojego monitora powinien wyglądać w przybliżeniu tak jak na rysunku 1. W głównym oknie programu znajdują się trzy tzw. okna dokowane.

- Okno oznaczone numerem 1 może składać się z kilku zakładek. Niewątpliwie najczęściej korzystać będziesz z zakładki Solution Explorer, która przedstawia strukturę załadowanej solucji w postaci hierarchicznego drzewa. Możesz wykorzystać je do przełączania się pomiędzy plikami projektów, utworzonych w ramach danej solucji.

Druga zakładka to ClassView. Przedstawia drzewo klas, struktur, globalnych zmiennych, funkcji oraz definicji. Zarówno tych utworzonych przez ciebie jak i obcych, które pojawiły się w projekcie po dołączeniu jakiegoś pliku z kodem. Często używaną zakładką jest także ResourceView. Pozwala na dodawanie i edytowanie zasobów programu. Zagadnieniem zasobów zajmowaliśmy się we wcześniejszych numerach WARP.




◄ **Rys. 1** Okno środowiska programistycznego Microsoft Visual Sdtudio 2005.

■ Okno Output (nr 2), to tzw. okno komunikatów. Dla ciebie, na początku będzie to okno, w którym Visual Studio będzie umieszczać informacje o błędach, ostrzeżeniach oraz sukcesach kompilacji i linkowania kodu aplikacji. Używane jest jednak też w wielu innych przypadkach, z którymi zetkniesz się w miarę postępującej pracy z Visual Studio.

■ Okno edytora (nr 3), służy oczywiście do edycji wielu różnych typów plików. Na rysunku 1 okno edytora przedstawia startową zakładkę Visual Studio. Każdy plik otwierany jest w osobnym oknie – zakładce.

## Tworzymy projekt aplikacji

Z menu ***File*** wybierz polecenie New, a następnie ***New Project.*** Na ekranie powinno ukazać się okno dialogowe o nazwie ***New Project***. Po lewej stronie znajduje się lista dostępnych projektów – ***Project Types***. Rozwiń korzeń ***Visual C++*** a następnie wyświetl zawartość pozycji ***Win32***. Okno powinna wyglądać teraz tak jak na **rysunku 2**.

Zatrzymajmy się na moment aby wyjaśnić pewną ważną sprawę. Praca w środowisku programistycznym ***Visual Studio 2005*** zorganizowana jest za pomocą solucji (ang. *solution*) i projektów (ang. *projects*). Zadaniem solucji jest grupowania projektów. W trakcie tworzenia projektu możemy wygenerować dla niego zupełnie nową solujcę lub dodać projekt do istniejącej solucji, która w danym momencie jest załadowana w środowisku Visual Studio. Do solucji możemy dodawać różne projekty, ich typ nie gra roli. Może być to projekt typu **Win32 Project**, a zaraz po nim, projekt typu **Win32 Console Applicatio**n, lub innego rodzaju. Kolejność dodawania projektów do solucji i ich typ nie ma znaczenia.

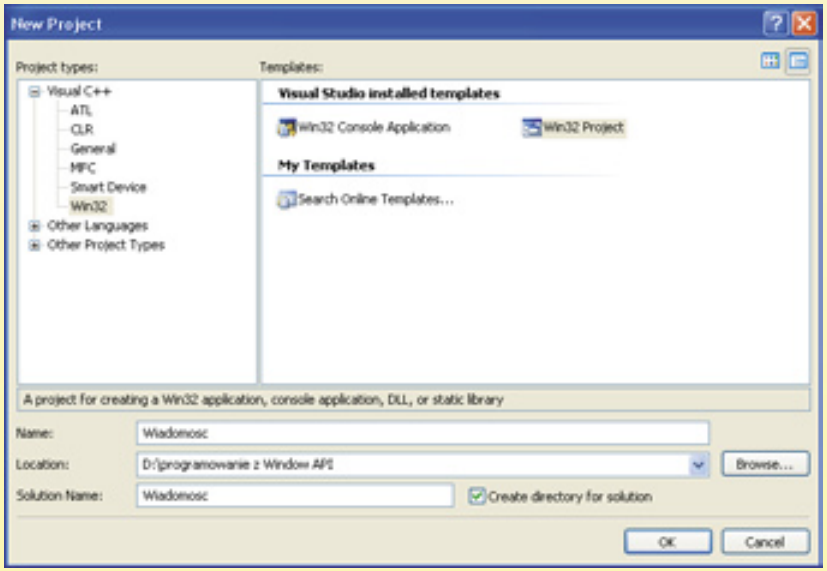
Podsumowując możemy powiedzieć, że solucje to pojemniki, które pozwalają grupować projekty. Solucja upraszcza równoległą pracę nad kilkoma projektami w ramach jednego zadania programistycznego. Wszystkie projekty z zakresu danej solucji znajdują się cały czas w zasięgu ręki, dzięki czemu bardzo szybko możemy się pomiędzy nimi przełączać i dokonywać różnych zmian.

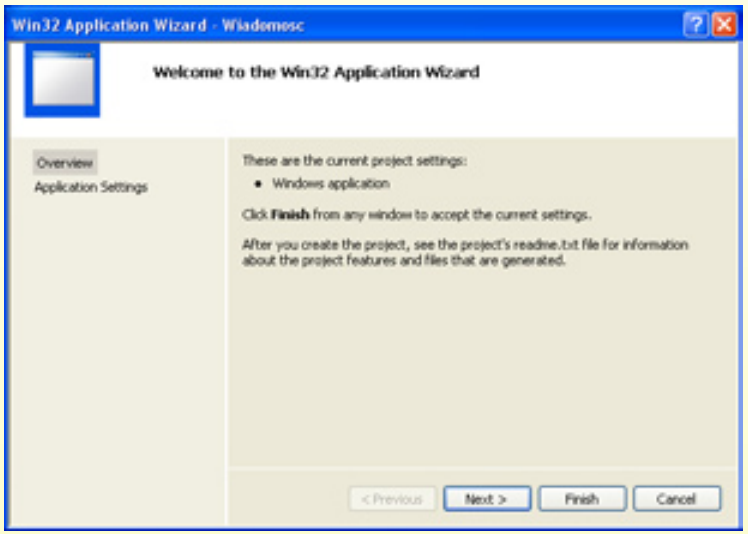
Przećwiczmy tworzenie solucji i projektu.

Na ekranie przed sobą powinieneś mieć wyświetlone okna dialogowe takie same jak na rysunku 2. Z listy szablonów projektów (ang. project temaplates) znajdującej się po prawej stronie, wybierz pozycję Win32 Project. Oznacza ona tworzenie projektu okienkowej aplikacji dla Windows.

Następnie u dału okna w polu Name wpisz wybraną przez siebie nazwę projektu. Proponuję, aby brzmiała

► **Rys. 2** Okno dialogowe ***New Project***.






◄ **Rys. 3** Okno dialogowe *Win32 Application Wizard*.

ona następująco: „Wiadomosc", wówczas łatwiej nam będzie dalej współpracować. Domyślnie ustalona przez ciebie nazwa projektu jest używana do określenia nazwy generowanego pliku wykonywalnego programu (*.exe). Oczywiście można to później zmienić w ustawieniach projektu. Zauważ także, że podawana nazwa jest automatycznie przepisywana do znajdującego się poniżej pola Solution name.

W polu Location należy podać ścieżkę do katalogu, w którym zostanie zapisany utworzony projekt.

Następnego pola – Solution name nie będziemy modyfikować, powiemy sobie tylko jaka jest jego rola. Możemy podać w nim indywidualną nazwę dla solucji w ramach, której zostanie utworzony projekt.

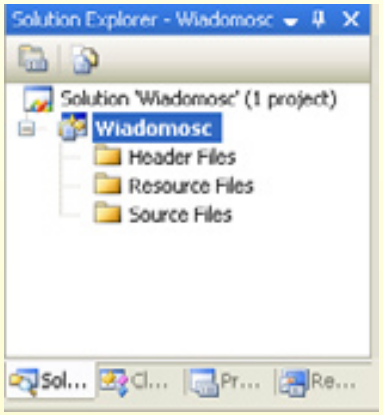
Przykładowo, jeżeli w ramach jednej solucji chciałbyś utworzyć dziesięć programów demonstrujących możliwości biblioteki OpenGL, dobrym rozwiązaniem byłby nadanie solucji osobnej nazwy, zbliżonej do „Możliwości OpenGL".

Jeżeli chcesz utworzyć katalog solucji, nadrzędny w stosunku do katalogów zawartych w niej projektów zaznacz opcję Create directory for solution. Dla powyższego przykładu mielibyśmy sytuację, w której wszystkie katalogi przykładowych programów OpenGL znajdowałby się w jednym zbiorczym katalogu „Możliwości OpenGL".

Po zaznaczeniu opcji Create directory for solution, upewnij się, że w wszystkie nazwy zostały podane prawidłowo, w szczególności nazwa projektu. Jeżeli wszystko jest w porządku naciśnij przycisk OK. Na ekranie pojawiło się nowe okno dialogowe o nazwie Win32 Application Wizard - Overview (rysunek 3).

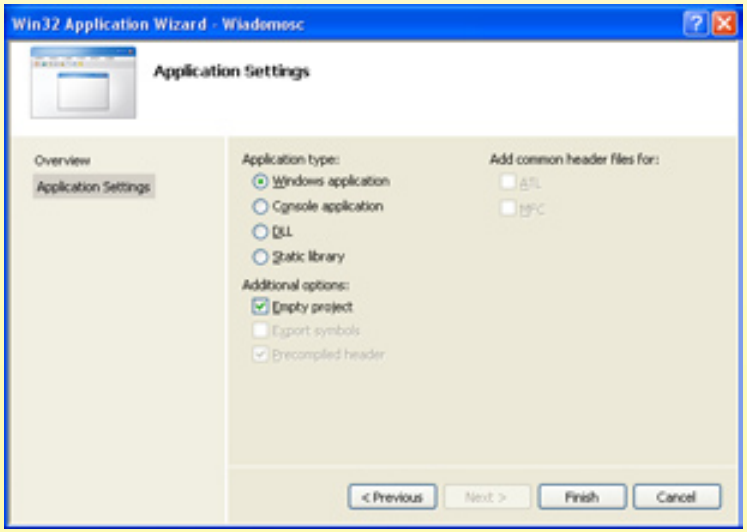
W tym oknie możemy zakończyć tworzenie projektu klikając przycisk Finish lub przejść do jego ustawień. Wybieramy to drugie, bowiem musimy dokonać drobnej korekty. Do ustawień przechodzimy klikając przycisk Next lub znajdujący się po lewej stronie odnośnik Application Settings. Po tej czynności zostanie wyświetlone okno takie jak na **rysunku 4**.

Lista Application Type pozwala ostatecznie określić, jaki typ projektu zamierzamy utworzyć. Może być to aplikacja okienkowa – Windows application, aplikacja konsolowa – Console Application, biblioteka łączna dynamicznie – Dynamic Link Library (DLL) lub biblioteka statyczna – Static library. Nas interesuje aplikacja okienkowa.




▲ **Rys. 5** Zakładka ***Solution Explorer*** solucji ***Wiadomosc***.

► **Rys. 4** Okno dialogowe ***Win32 Application Wizard – Application Settings***.



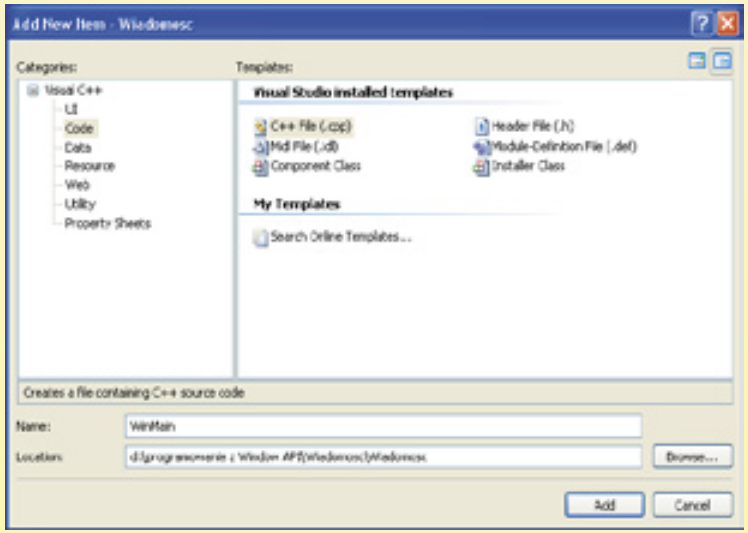
Na liście dodatkowych opcji – Additional Options zaznaczmy tworzenie pustego projektu wybierając opcję Empty Project. W innym przypadku Visual Studio wygeneruje dla nas projekt zawierający kod standardowej windowsowej aplikacji, wyświetlającej pojedyncze okno. W pewnych sytuacjach, kiedy nabierzesz już nieco wprawy w posługiwaniu się Windows API, ta opcja może okazać dość pomocna. Aktualnie jednak chcemy utworzyć całkowicie pusty projekt aplikacji, nie zawierający żadnych plików. Po zaznaczeniu opcji Empty Project naciśnij przycisk Finish.

W tym momencie folder projektu o nazwie zgodnej z nazwą projektu, został utworzony w katalogu solucji Wiadomosc. Znajduje się w nim plik projektu o nazwie Wiadomosc.vcproj. Na zakładce Solution Explorer widzimy, że solucja zawiera teraz jeden projekt o nazwie Wiadomosc. Do ikony projektu podczepione są trzy foldery: Source Files (pliki źródłowe), Header Files (pliki nagłówkowe) i Resource Files (pliki zasobu). Spójrz na **rysunek 5**. Wszystkie foldery są puste. Oczywiście, dlatego że świadomie utworzyliśmy pusty projekt.

## Dodajemy do projektu plik kodu

Teraz do projektu dodamy formularz / plik kodu C++. W tym celu rozwiń menu Project i wybierz polecenie ***Add new item***. W wyświetlonym oknie (**rysunek 6**), wybierz kategorię ***Code***. Po prawej stronie z listy dokumentów do utworzenia, wybierz pozycję ***C++ File***. Następnie poniżej, w polu Name wpisz nazwę jaką ma przybrać tworzony plik. Jak się z pewnością domyślasz także i tym razem Ci ją zasugeruję. Niech brzmi następująco: „***WinMain***”. Dlaczego akurat tak a nie inaczej, dowiesz się niebawem. Pole ***Location*** pozostawiamy bez zmian. Znajduje się w nim ścieżka lokalizująca plik. Zostaje ustalana automatycznie na podstawie ścieżki do katalogu projektu.

Teraz pozostaje już tylko kliknąć przycisk Add, a plik ***WinMain.cpp*** zostanie utworzony i dodany do projektu ***Wiadomość***.




► **Rys. 6** Okno Add new item.

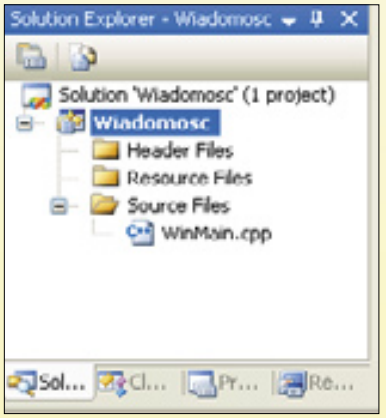
## Piszemy kod programu

Plik ***WinMain.cpp*** został utworzony i jednocześnie pojawiło się jego okno edycji. Zerknij do katalogu ***Source Files*** projektu Wiadomość na zakładce ***Solution Explorer*** (**rysunek 7**). Jak widzisz, znajduje się tam graficzna reprezentacja pliku ***WinMain.cpp***. Dwukrotne kliknięcie na ikonę pliku powoduje wyświetlenie jego zawartości. Przepisz teraz do pliku ***WinMain.cpp*** kod z **listingu 1**.

Jest to kod aplikacji, która po uruchomieniu wyświetli okienko z zapytaniem „*Witam! Czego chcesz?*”. Dokładnie takie samo jak na **rysunku 8**.

Po naciśnięciu w okienku przycisku ***OK*** program kończy swoje działanie.

Główną funkcją aplikacji windowsowej, od której rozpoczyna się działanie programu jest funkcja ***WinMain***. Właśnie z tego powodu, utworzonemu wcześniej formularzowi kodu nadaliśmy nazwę ***WinMain***. Główna korzyść, jaka z tego płynie jest następująca: Jeżeli projekt będzie składał się z powiedzmy np. 30 plików i każdy z nich będzie miał odmienną nazwę to zaglądając do



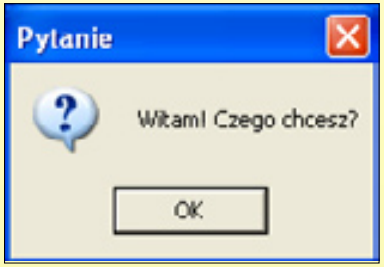

▲ **Rys. 7** Zakładka ***Solution Explorer*** po dodaniu pliku ***WinMain.cpp***.

katalogu projektu będziemy mogli bardzo szybko odszukać ten właściwy plik, zawierający definicję głównej funkcji programu. Oczywiście można to zrobić równie sprawnie i bez tego. Wystarczy po prostu załadować projekt do Visual Studio i odnaleźć globalną funkcję WinMain na zakładce **Class View**.

Przeanalizujmy krok po kroku kod programu. Zrobimy to bardzo dokładnie, jeżeli czegoś nie będziesz rozumiał po prostu nie przejmuj się tym. Analizujemy drobiazgowo, chcę bowiem abyś wiedział dokładnie co oznaczają poszczególne fragmenty kodu. Pierwszy wierz ma postać:

```
#include <windows.h>
```

Jest to dyrektywa preprocesora, która powoduje włączenie zawartości pliku nagłówkowego windows.h do pliku WinMain.cpp. Plik windows.h zawiera definicje wielu identyfikatorów i deklaracje funkcji Windows API. Sam plik windows.h posiada szereg własnych dyrektyw preprocesora włączających kolejne pliki nagłówkowe. Oto nazwy kilki plików nagłówkowych, które są włączane do pliku windows.h. Skąd później razem z nim wędrują do pliku WinMain. cpp: windef.h, winbase.h, wingdi.h, winuser.h.




▲ **Rys. 8** Zakładka ***Solution Explorer*** po dodaniu pliku ***WinMain.cpp***.

Działanie programu rozpoczyna się od funkcji WinMain. Przyjrzyjmy się bliżej jej deklaracji.

```
int WINAPI WinMain(
  HINSTANCE hInstance,
  HINSTANCE hPrevInstance,
  LPSTR lpCmdLine,
  int nShowCmd);
```

Funkcja **WinMain** zwraca wartość typu **int**. Jak wiemy rozmiar typu int jest zależny od systemu. Dla jednych wynosi 2 a dla innych 4 bajty. W przypadku systemu Windows są to 4 bajty. Zmienne typu int mogą przechowywać wartości dodatnie i ujemne. Za znak przechowywanej wartości odpowiada stan najbardziej znaczącego bitu. Jednak to nie to jest teraz dla nas najważniejsze.

Zwracanie wartości typu int przez funkcję WinMain oznacza, że obok instrukcji return, musimy postawić pewną wartość zgodną z typem int. Zadaniem return jest zakończenie działania funkcji i zwrócenie sterowania do funkcji wywołującej. W przypadku WinMain ową wywołującą funkcją jest system. My obok return postawimy zero, co oznacza zwrócenie do systemu informacji, że program zakończył swoje działanie poprawnie. Chyba nie muszę tłumaczyć, że opuszczenie funkcji WinMain, jest równoznaczne z zakończeniem pracy programu.

**Listing WIN-1.1** Wyświetlenie okna komunikatu. (projekt: Wiadomość, plik: WinMain.cpp)

```
01: // P L I K I    N A G Ł Ó W K O W E
02: ////////////////////////////////////////
03: #include <windows.h>
04:
05: // W I N M A I N
06: ////////////////////////////////////////
07: int WINAPI WinMain(
08:   HINSTANCE hInstance, HINSTANCE hPrevInstance,
09:   LPSTR lpCmdLine, int nShowCmd)
10: {
11:
12:   MessageBox(0, "Witam! Czego chcesz?",
13:     "Pytanie", MB_OK | MB_ICONQUESTION);
14:
15: return 0;
16: }
```

Przesuwając się w prawo po deklaracji funkcji WinMain widzimy, że obok typu wartości zwracanej (int) stoi identyfikator WINAPI. Jest on zdefiniowany w pliku nagłówkowym Windef.h w następujący sposób:

```
#define WINAPI    __stdcall
```

Czyli widzimy, że podczas kompilacji programu na etapie pracy preprocesora, identyfikator **WINAPI** zastępowany jest słowem **__stdcall**. To słowo kluczowe oznacza konwencję wywołania funkcji. Jego zastosowanie oznacza odkładanie na stosie parametrów funkcji przed jej wywołaniem w kolejności od prawej do lewej strony. Zmusza również funkcję do zdjęcia własnych parametrów ze stosu przed zakończeniem działania.

Analizując dalszą część deklaracji funkcji **WinMain**, napotykamy identyfikator WinMain, jest to po prostu nazwa głównej funkcji programu windowsowego. Jednak funkcja WinMain nie do końca jest taką sobie zwyczajną funkcją. Jej niezwykłość polega na tym, że nie może zostać przeładowana, tzn. że w kodzie programu może się znaleźć tylko jedna definicja funkcji o nazwie WinMain.

Zajmijmy się teraz parametrami funkcji WinMain. Pierwszy parametr – ***hInstance***, jest typu HINSTANCE. Jest to tzw. uchwyt programu. Uchwyt to po prostu jakiś numer do oznaczania czegoś. W tym przypadku parametrowi hInstance wartość nada system, który uruchamiając aplikacje przypisze mu jakiś numer (oznaczenie). Uchwyt aplikacji

przechowuje się na ogół w jakiejś zmiennej globalnej. Wymagany jest przez niektóre funkcje Windows API.

Drugi parametr – ***hPrevInstance*** jest identycznego typu jak pierwszy. Miał on swoje znaczenie w 16-bitowych aplikacjach, dla starszych wersji Windows. Oznaczał uchwyt poprzedniej realizacji programu. Jeżeli uruchomimy kilka kopii programu, tworzymy pewną liczbę jego realizacji. W 32-aplikacjach Windows, czyli również i w naszej, drugi parametr nie jest używany i zawsze przybiera wartość zero.

Trzeci parametr funkcji WinMain - ***lpCmdLine***, jest wskaźnikiem do łańcucha znaków tzw. wiersza poleceń lub linii wywołania aplikacji. Czasem przydaje się do pobrania od użytkownika pewnych startowych parametrów dla programu.

Czwarty parametr – ***nShowCmd*** jest typu int i określa sposób wyświetlenia aplikacji. Jak wiesz aplikacja Windows może zostać wyświetlona w postaci zminimalizowanej, zmaksymalizowanej lub normalnej. Domyślnie parametr nShowCmd przyjmuje wartość 1 (SW_ SHOWNORMAL) co oznacza standardowe (normalne) wyświetlenie aplikacji. Wartość parametru nShowCmd może zostać umyślnie zmieniona podczas jawnego uruchomienia danego programu przez inny program za pomocą funkcji **WinExec** lub **CreateProcess**. Zajmowaliśmy się nimi w jednym z pierwszych WARP-ów.

Skończyliśmy omawiać nagłówek funkcji WinMain, dalej zajmiemy się jej ciałem, czyli kodem zawartym między nawiasami **{ }**. W ciele znajduje się wywołaniem funkcji **MessageBox**.

## Funkcja MessageBox

Funkcja WinMain naszego programu zawiera tylko dwie instrukcje: wywołanie funkcji **MessageBox** oraz return. Znaczenie instrukcji return omówiłem już wcześniej teraz powiem kilka słów o funkcji MessageBox. Jak już się dowiedziałeś na samym początku, funkcja MessageBox służy do wyświetlania okienka z komunikatem. Jej deklaracja ma postać:

```
int MessageBox(
  HWND hWnd,
  LPCTSTR lpText,
  LPCTSTR lpCaption,
  UINT uType);
```

Jako parametr ***hWnd*** podajemy uchwyt okna, które stanie się właścicielem okienka komunikatu. Możliwe jest również podanie wartości 0. Wówczas okienko komunikatu nie będzie miało właściciela. Użytkownik nie może wrócić do pracy w oknie, które jest właścicielem wyświetlonego okna komunikatu, dopóki nie odpowie na zawarty w nim komunikat. Odpowiedź polega po prostu na naciśnięciu któregoś z przycisków okienka.

Drugi parametr ***lpText*** to wskaźnik do łańcucha znaków zakończonego zerem. Stanowić on będzie treść komunikatu.

Trzeci parametr ***lpCaption*** to również wskaźnik do łańcucha znaków zakończonego zerem. Stanowić będzie treść napisu na belce okienka komunikatu.

Czwarty parametr ***uType*** określa jakie przyciski, ikony i dodatkowe opcje mają zostać dodane do okna komunikatu. Wartość tego parametru może stanowić kombinacja kilkunastu flag.

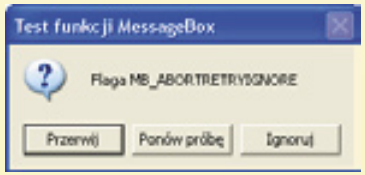
Wszystkie flagi funkcji ***MessageBox*** rozpoczynają się od przedrostka MB. Poniżej został przedstawiony pierwszy zestaw flag, odpowiedzialny za rodzaj wyświetlanych przycisków. Należy zdecydować się tylko na jedną flagę przycisków. Nie dotyczy to flagi MB_HELP, która może zostać użyta z wszystkimi pozostałymi. Jej zadanie polega na dodaniu przycisku ***Pomoc*** do wybranego zestawu przycisków.

Oto lista flag przycisków oraz przykładowych okienek wiadomości z uzyskanym efektem.

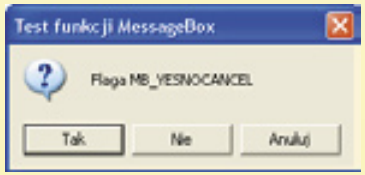
MB_OK


MB_OKCANCEL


MB_ABORTRETRYIGNORE

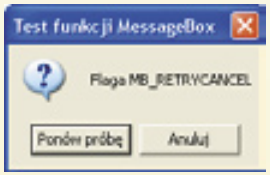



MB_YESNOCANCEL




MB_YESNO

MB_RETRYCANCEL




MB_HELP (przykład użycia z flagą MB_OK)


Poniżej znajduje się zestaw flag odpowiedzialnych za rodzaj wyświetlanej ikony. Niektóre z flag mają swoje alternatywne odpowiedniki.

MB_ICONHAND
MB_ICONERROR
MB_ICONSTOP


MB_ICONQUESTION


MB_ICONEXCLAMATION
MB_ICONWARNING


MB_ICONINFORMATION
MB_ICONASTERISK


Następujące flagi są odpowiedzialne za wybranie domyślnego przycisku okna komunikatu. Domyślny przycisk to ten, który reaguje na wduszenie klawisza Enter lub Space. Wyróżnia się tym, że zaraz po wyświetleniu okienka posiada nieco pogrubioną obwódkę.

**MB_DEFBUTTON1**
pierwszy przycisk jest domyślnie wybrany

**MB_DEFBUTTON2**
drugi przycisk jest domyślnie wybrany

**MB_DEFBUTTON3**
trzeci przycisk jest domyślnie wybrany

**MB_DEFBUTTON4**
czwarty przycisk jest domyślnie wybrany

Oprócz flag określających rodzaj wyświetlanych przycisków i ikon do dyspozycji mamy dodatkową grupę flag określających sposób działania okienka komunikatu. Podam tylko jedną z nich.

**MB_TOPMOST**
okno komunikatu pozostaje zawsze na wierzchu, nawet gdy przełączymy się do innej aplikacji.

To by było na tyle, jeżeli chodzi o flagi tworzące czwarty parametr funkcji MessageBox. Zajmijmy się wartością zwracaną przez funkcję. Jest ona typu int i może być równa jednej z podanych niżej stałych. Zwrócenie wartości następuje po zamknięciu okna komunikatu.

**IDOK**
został wybrany (naciśnięty) przycisk OK.

**IDCANCEL**
został wybrany przycisk Anuluj.

**IDABORT**
został wybrany przycisk Przerwij.

**IDRETRY**
został wybrany przycisk Ponów próbę.

**IDIGNORE**
został wybrany przycisk Zignoruj.

**IDYES**
został wybrany przycisk Tak.

**IDNO**
został wybrany przycisk Nie.

Wartość IDCANCEL zastaje zwrócona również w sytuacji, gdy okienko komunikatu posiada przycisk Anuluj i zostanie naciśnięty klawisz Esc. Gdy okno komunikatu nie posiada przycisku Anuluj, naciśnięcie klawisza Esc nie powoduje żadnego działania.

Może Cię dziwić to, że nie ma wartości zwracanej informującej o naciśnięciu przycisku Pomoc (dodanego za pomocą flagi MB_HELP). Naciśnięcie przycisku Pomoc nie powoduje zamknięcia okna komunikatu, a jedynie wysłanie do okna rodzica (właściciela okna komunikatu) wiadomości WM_HELP, która powinna zostać odpowiednio obsłużona. Tematem wysyłania i odbierania wiadomości zajmiemy się nieco dalej.

Funkcja MessageBox potrafi również zwrócić wartość 0, gdy system posiada zbyt mało pamięci do utworzenia struktur okienka komunikatu i jego wyświetlenia.

## Kompilujemy kod i uruchamiamy program

Najwyższa pora utworzyć plik wykonywalny (*.exe) projektu Wiadomosc. W tym celu musimy poddać projekt tzw. budowaniu, na które składa się kompilowanie i linkowanie (konsolidacja) jego kodu.

Konsolidacji nie podlega wyłącznie kod projektu, w tej operacji do programu dołączany jest także kod bibliotek, z których programista skorzystał.

Aby skompilować i jednocześnie zlinkować cały kod programu rozwiń menu Build i wybierz opcję Build, przy której będzie dopisana nazwa projektu. W naszym przypadku jest do dopisek „Wiadomosc". Podobne działanie możemy wykonać naciskając klawisz F7. Różnica polega jednak na tym, że kompilacji i linkowaniu poddawana jest cała solucja. Zatem jeżeli w skład solucji wchodzi więcej niż jeden projekt, nastąpi kompilacja i linkowanie każdego z nich. Jeżeli w kodzie projektu od ostatniej kompilacji nie dokonano żadnych modyfikacji, kompilacja nie zostanie wykonana.

W trakcie budowy pliku Wiadomosc.exe okno Output będzie wypełniać się kolejnymi komunikatami. Pierwsza ich grupa dotyczy etapu kompilacji plików CPP. Projekt Wiadomosc zawiera tylko jeden taki plik – WinMain.cpp. I tylko on zostanie skompilowany do kodu obiektowego, któremu będzie odpowiadał plik WinMain.obj. Jeżeli przepisując kod programu nie popełniłeś błędu, w oknie Output po etapie kompilacji powinny pojawić się tylko poniższe logi:

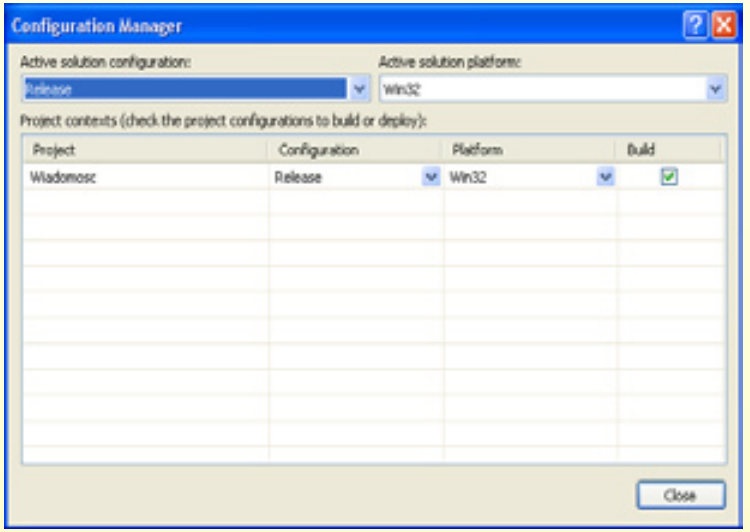
```
Compiling...
WinMain.cpp
```

Jeżeli Twoje okno Output zawiera dodatkowe komunikaty o wystąpieniu błędów (ang. error) lub ostrzeżeń (ang. warning), sprawdź czy przepisany przez ciebie kod programu jest zgodny z listingiem 1. Jeżeli wykryjesz błędy, usuń je i powtórz czynność budowy pliku wykonywalnego.

Błędy wynikłe podczas kompilacji nie pozwalają na dalsze działanie linkera, cały proces budowy pliku wykonywalnego zostanie przerwany. Z ostrzeżeniami rzecz ma się nieco inaczej. Czynność kompilacji i linkowania będzie kontynuowana, jednak utworzony program może w niektórych przypadkach działać niepoprawnie. Mogą pojawić się tzw. logiczne błędy uruchomienia.

Kompilator nie zawsze wykrywa wszystkie popełnione przez nas błędy logiczne, dlatego są one o wiele groźniejsze od błędów kompilacji, gdyż to my samodzielnie musimy je wykryć i usunąć. A nierzadko jest to bardzo trudne zadanie.

Gdy etap kompilacji zakończy się pomyślnie do pracy przystępuje program łączący (linker). Jego zadaniem jest dołączenie do programu kodu bibliotecznego oraz połączenie całego obiektowego kodu, powstałego podczas kompilacji plików wchodzących w skład projektu.




► **Rys. 9** Okno Configuration Manager, w którym możemy dokonać zmiany aktywnej konfiguracji projektu.

Etap linkowania rozpoczyna się z chwilą, gdy w oknie Output pojawi się następujący komunikat:

```
Linking...
```

Jeżeli pojawią się pod nim dodatkowe wpisy mówiące o błędach, należy je usunąć i ponownie wykonać operację budowy pliku wykonywalnego.

Jakie błędy mogą wystąpić podczas konsolidacji kodu programu? Powodów ich pojawienia się może być bardzo wiele. Problem wystąpi gdy linker nie będzie mógł odnaleźć kodu funkcji bibliotecznej użytej w programie. To się zdaża, gdy nie dołączymy odpowiednich bibliotek do projektu. Problem wystąpi również, gdy źle podamy nazwę biblioteki. Linker będzie po prostu próbował otworzyć nieistniejący plik.

Kłopoty napotkamy także, gdy program łączący będzie próbował zbudować plik wykonywalny a ten będzie już istniał i w dodatku będzie zabezpieczony przed zapisem tzn. będzie posiadał ustawioną opcję Tylko do odczytu (atrybuty pliku możemy sprawdzić klikając na jego ikonie prawym przyciskiem myszy i wybierając opcję Właściwości). Do takiej sytuacji często dochodzi gdy próbujemy kompilować projekt przeniesiony z płyty CD na twardy dysk. Wszystkie pliki na płycie CD są przeznaczone wyłącznie do odczytu i w takiej samej postaci są przenoszone na dysk twardy. Aby poradzić sobie z tym problemem wystarczy, że wyłączymy opcję Tylko do odczytu dla wszystkich plików, które mogą

w procesie kompilacji i linkowania podlegać ponownemu tworzeniu lub modyfikacji.

Prawie wszystkie komunikaty kompilatora i linkera o błędach i ostrzeżeniach są na tyle czytelne i zrozumiałe, że z ich lokalizowaniem i usuwaniem nie powinieneś mieć żadnych problemów. Każdy błąd i ostrzeżenie oznaczony jest specjalnym kodem. Jego opis można znaleźć w bibliotece MSDN (ang. Microsoft Developer Network Library).

Jeżeli proces kompilacji i łączenia kodu zakończy się pomyślnie otrzymasz plik wykonywalny projektu Wiadomosc – Wiadomosc.exe. Cały tekst okna Output powinien wyglądać następująco (usunąłem z niego zbędne wpisy):

```
-- Build started: Project:
Wiadomosc, Configuration:
Debug Win32 --
Compiling...
WinMain.cpp
Linking...
Wiadomosc - 0 error(s), 0
warning(s)
==== Build:
 1 succeeded,
 0 failed,
 0 up-to-date,
 0 skipped ====
```

Jak widać powyżej, na końcu umieszczane jest podsumowanie określające liczbę wykrytych błędów i ostrzeżeń na etapie kompilacji i konsolidacji dla danego projektu. Wyświetlane jest także drugie podsumowanie odnoszące się do każdej konfiguracji projektu, która została poddana budowie.

Aby uruchomić utworzony program rozwiń menu Debug i wybierz opcję Start Without debugging. Tą samą czynność możesz wykonać naciskając kombinację klawiszy Ctrl+F5.

## Więcej o Visual Studio

Poniżej podam kilka dodatkowych informacji, które pomogą ci lepiej zrozumieć działanie środowiska programistycznego Visual Studio 2005.

## Konfiguracja Debug i Release

W poprzednim podpunkcie utworzyliśmy plik wykonywalny projektu Wiadomosc. Zajrzyj do katalogu projektu. Znajduje się tam folder o nazwie Debug a w nim plik Wiadomosc.exe. Po sprawdzeniu jego rozmiaru możesz się odrobinę zdziwić. Program w zasadzie nic nie robi, wyświetla tylko małe okienko, a jego plik zajmuje około 40KB przestrzeni dyskowej. Zastanawiasz się pewnie czy plik wykonywalny, którego cały kod składa się zaledwie z kilku instrukcji powinien być aż tak duży. Rozwiązanie tej zagadki jest dość proste. Pamiętasz wydruk w oknie Output po poprawnym skompilowaniu i zlinkowaniu kodu projektu Wiadmosc? Pierwszy wiersz zwierał słowa:

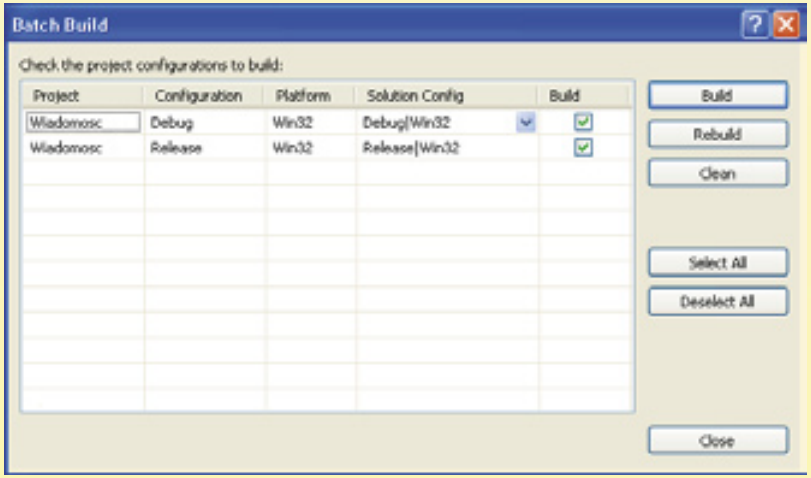
```
Configuration: Debug Win32
```

Zwróć szczególną uwagę na słowo Configuration i Debug. Już wyjaśniam o co chodzi. Włączona konfiguracja Debug oznacza wstawienie do tworzonego pliku wykonywalnego dodatkowego kodu. Ułatwia on śledzenie poprawności działania programu. Korzysta z niego głównie Debugger. Z konfiguracji Debug powinieneś korzystać w trakcie tworzenia i testowania aplikacji. A gdy projekt zostanie ukończony i kod będzie gotowy do ostatecznej kompilacji przed oddaniem aplikacji w ręce użytkownika, konfigurację Debug powinniśmy zmienić na konfigurację Release. Druga nie obciąża programu dodatkowym kodem, który był wymagany przy testowaniu aplikacji.

Konfiguracja Debug i Release to dwie podstawowe konfiguracje dostępne w środowisku Visual Studio. Mamy także możliwość tworzenia własnych, dodatkowych konfiguracji. Każda z nich może określać odmienne ustawienia kompilatora i linkera, przy uwzględnieniu których zostanie zbudowany plik wynikowy projektu.

Aby zmienić konfigurację projektu z Debug na Release rozwiń menu Build i wybierz opcję Configuration Manager. Zostanie wyświetlone okno o takiej samej nazwie (rysunek 9).

Rozwijana lista Active solution configuration pozwala ustalać konfigurację jednocześnie dla wszystkich projektów solucji. Natomiast jeżeli chcemy dla danego projektu wybrać konfigurację indywidualnie, musimy skorzystać z listy znajdującej się poniżej. Dokonane zmiany w ustawieniach zatwierdzamy




► **Rys. 10** Okno ***Batch Build***. Umożliwia jednoczesną budowę projektów w różnych konfiguracjach.

klikając przycisk **Close**.

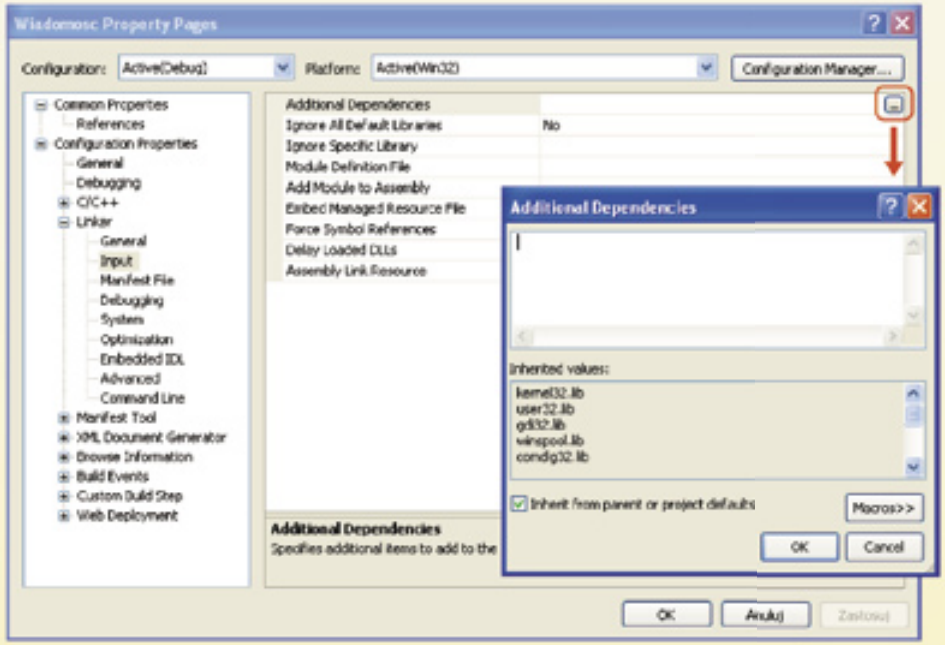
Po zmianie konfiguracji na Release należy ponownie skompilować i zlinkować kod projektu Wiadomosc. W jego katalogu zostanie utworzony nowy folder o nazwie Release, w którym znajdzie się nowy plik Wiadomosc.exe nie zawierający kodu debugującego. Możesz sprawdzić, że rzeczywiście rozmiar pliku jest dużo mniejszy od jego odpowiednika z katalogu Debug.

Visual Studio umożliwia także jednoczesną budowę projektów w różnych konfiguracjach. Należy z menu Build wybrać opcję **Batch Build**. W wyświetlonym oknie (**rysunek 10**) zaznaczmy, które konfiguracje chcemy poddać jednej z trzech operacji.

Może być to zwyczajna budowa (przycisk **Build**) oznaczająca kompilację i konsolidację tylko w przypadku, kiedy w projekcie wykryte zostaną jakieś zmiany. Ponowna budowa (przycisk **Rebuild**) oznaczająca usunięcie i ponowne utworzenie wszystkich plików powstałych podczas kompilacji i konsolidacji. Wyczyszczenie projektu (przycisk **Clean**) oznaczające usunięcie wszystkich plików będących produktami kompilacji i konsolidacji.

## Biblioteki statyczne

Gdy mówiliśmy jak skompilować i skonsolidować kod programu wspomniałem, że w trakcie operacji linkowania, do pliku wykonywalnego powinien zostać dodany kod bibliotek, które programista wykorzystał podczas tworzenia aplikacji. W projekcie Wiadomosc skorzystaliśmy z funkcji MessageBox. Jej kod znajduje się w bibliotece user32.lib (lib – library). Ta biblioteka do projektu typu Win32 Project jest dodawana automatycznie. Zobaczmy teraz jak własnoręcznie dodać do projektu takie biblioteki.




▲ **Rys. 11**
Okno właściwości projektu. Pozwala dokonywać zmian w ustawieniach kompilatora i linkera dla danej konfiguracji projektu.

Rozwiń menu **Project** i wybierz opcję **Properties**. Zostanie wyświetlone okno dialogowe widoczne na **rysunku 11**. Korzystając ze znajdującego się po lewej stronie drzewa przejdź do parametrów wejściowych linkera (**Configuration Properties>Linker> Input**). Nazwy bibliotek statycznych, które chcemy dołączyć do projektu wpisujemy w widocznym po prawej stronie polu Additional Dependencies. Kiedy może zajść taka potrzeba? Przykładowo wówczas, kiedy przy tworzeniu programu korzystasz z jakiś dodatkowych API (Application Programming Interface), np. graficznej biblioteki DirectX lub OpenGL. Czasami niektóre funkcje Windows API także wymagają dołączenia kolejnych bibliotek. Skąd jednak masz wiedzieć, z jakiej biblioteki pochodzi dana funkcja? Dokumentacja, w której funkcja została opisana zazwyczaj podaje takie informacje.

Listę bibliotek domyślnie dodawanych do projektu, możesz wyświetlić naciskając przycisk znajdujący się po prawej stronie pola **Additional Dependencies**.

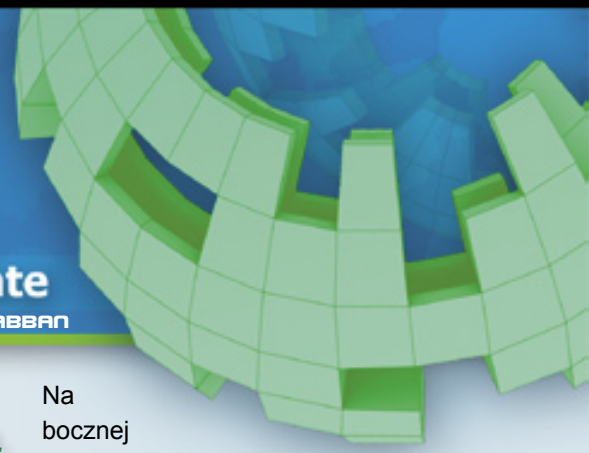
Wiesz już jak do konsolidacji projektu dodać bibliotekę statyczną. Teraz zobaczymy gdzie są podane ścieżki katalogów bibliotek. Visual Studio musi znać ścieżkę dostępu, aby linker mógł pobrać bibliotekę, kiedy będzie jej potrzebował.

Rozwiń menu Tools i wybierz opcję Options. Zostanie wyświetlone okno widoczne na **rysunku 12**. Przejdź do zawartości **Projects and Solutions>VC++ Directories**. Z listy **Show directories** for wybierz pozycję **Library files**. Poniżej zostaną wyświetlone ścieżki do katalogów, w których znajdują się pliki bibliotek. Możesz dodawać własne ścieżki i usuwać już istniejące. ■

# 3ds max® Modyfikacja HSDS

## Zastosowanie HSDS oraz podział za pomocą Tessellate

DANIEL BESTA A.K.A. RABBAN



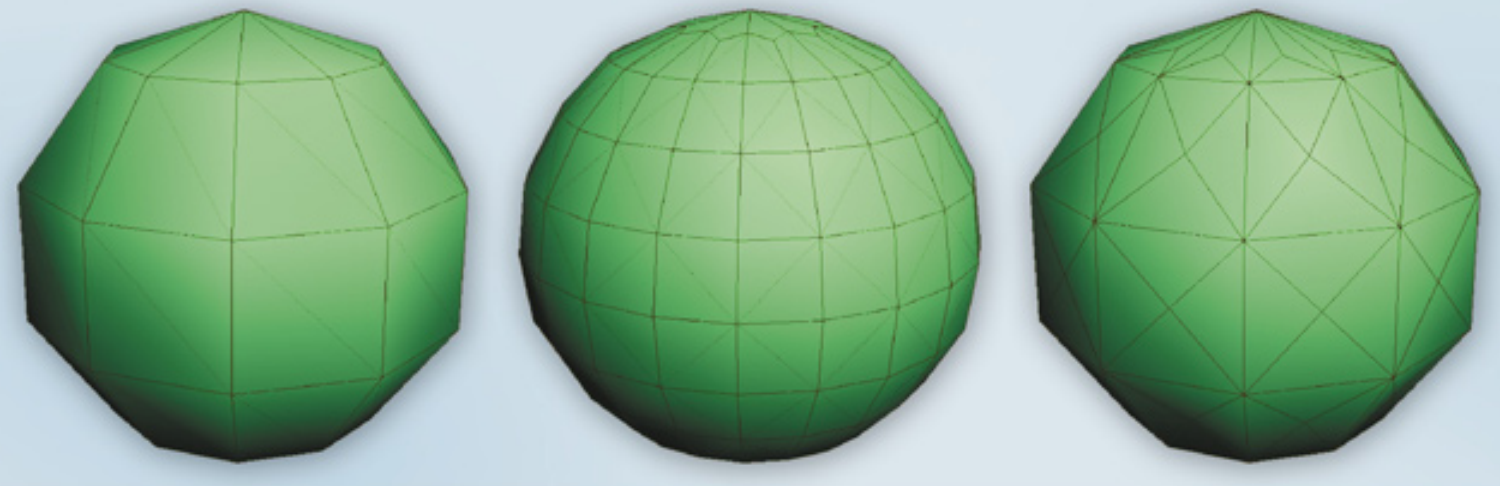
*Temat modelowania obiektów w 3ds max poruszaliśmy już wielokrotnie. W naszym magazynie pisaliśmy o modyfikacjach Edit Mesh, Edit Poly, czy też metodzie Nurms. Tym razem zajmiemy się modyfikacją HSDS, która pozwala przyspieszyć pracę nad detalami i ułatwia proces podziału wybranych fragmentów siatki. Omówimy również narzędzie Tessellate, inną modyfikację dzielącą ścianki.*

W trakcie modelowania obiektu bardzo często stajemy przed koniecznością dodatkowego podziału siatki. Powodem może być chęć poprawienia kształtu obiektu lub zamiar dodania mniejszych detali w konstrukcji modelu. W 3ds max możemy zwiększać gęstość siatki zarówno całego obiektu, jak i tylko jego wybranego fragmentu.

### Tessellate

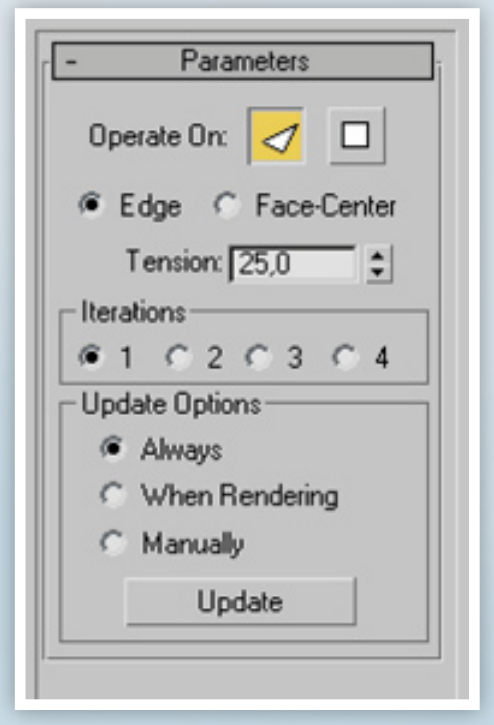
Jeżeli mamy zamiar zwiększyć gęstość całej siatki, to jednym z dostępnych rozwiązań jest zastosowanie modyfikacji **Tessellate**. Jest to bardzo proste w obsłudze narzędzie, które posiada zaledwie kilka łatwych do opanowania opcji.

Po przypisaniu modyfikacji do obiektu siatka automatycznie zostaje podzielona. Na bocznej listwie znajdziemy zestaw opcji w rolecie **Parameters**, który pozwala określić sposób i stopień podziału. Możemy tutaj wybrać rodzaj ścianek – przy włączonej opcji **Face** pracujemy z siatką zbudowaną z trójkątów, natomiast dla opcji **Polygon** powstaje siatka złożona z wieloboków. Opcja wyboru **Edge** i **Face-Center** decyduje o sposobie podziału ścianek. W obu przypadkach ścianki dzielone są poprzez wstawienie w ich środek nowego wierzchołka, jednocześnie powstają również krawędzie łączące.



◄◄◄ **Tak oto funkcjonuje modyfikacja Tessellate. Widzimy tutaj trzy kule, pierwszy obiekt z lewej jest to początkowa, podstawowa 10-segmentowa kula Sphere. Do tego obiektu przypisujemy modyfikację Tessellate - dwie kule po prawej ukazują efekty. Dla środkowej kuli zastosowano modyfikację Tessellate i dokonano podziału metodą Edge. Natomiast kula po prawej stronie ukazuje podział metodą Face-Center. Warto zauważyć, że metoda Edge powoduje wygładzenie, większe zaokrąglenie obiektu, gdyż krawędzie dzielone są na pół. W przypadku opcji Face-Center następuje jedynie zagęszczenie siatki, metoda dodaje nowe krawędzie ale nie wygładza powierzchni.**

Dla opcji Edge są to krawędzie łączące środek ścianki ze środkami jej krawędzi, natomiast dla opcji Face-Center są to krawędzie łączące środek ścianki z jej pozostałymi wierzchołkami. Nieco



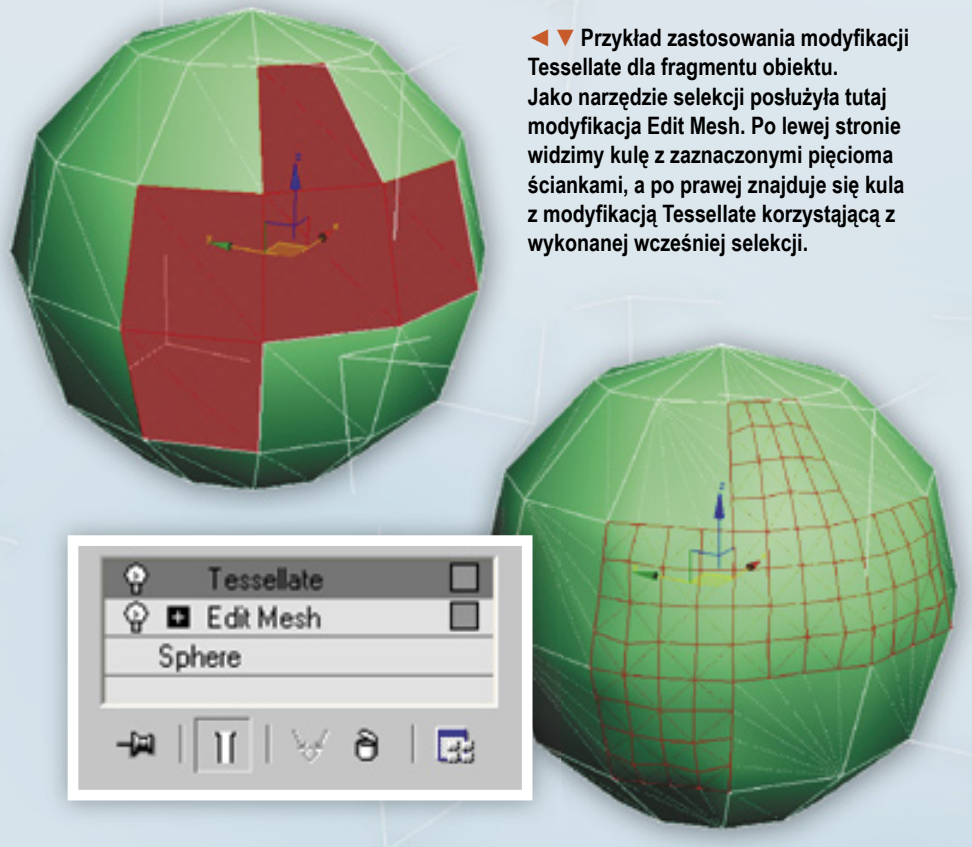

▲ **Zestaw opcji modyfikacji Tessellate.**

niżej znajduje się parametr Tension, który pozwala wypychać lub wciskać utworzone w trakcie podziału wierzchołki.

W grupie **Iteration** znajdują się cztery opcje wyboru określające ilość powtórzeń operacji Tessellate. Domyślnie włączone jest jedno powtórzenie Iteration, dlatego też ścianki podzielone są raz. Przykładowo wybranie wartości 2 powoduje, że operacja podziału wykonana jest dwukrotnie – ten sam efekt osiągniemy przypisując dwie modyfikacje Tessellate o wartości Iteration równej 1. Na samym dole znajdują się opcje **Update Options** określające sposób uaktualniania zmian wprowadzanych w opcjach modyfikacji.

Modyfikacja Tessellate może być również wykorzystywana do dzielenia wybranych fragmentów siatki. W tym wypadku skorzystać musimy z dodatkowej modyfikacji, która pozwoli nam na dokonanie odpowiedniej selekcji. Dla przykładu może to być **Edit Mesh**. Jeżeli zaznaczymy fragment powierzchni obiektu za pomocą modyfikacji Edit Mesh, a następnie przypiszemy modyfikację Tessellate, to podział będzie dotyczył tylko aktualnego zestawu selekcji.

Zwiększanie gęstości tylko wybranych fragmentów siatki jest bardzo przydatne podczas modelowania obiektów. Przykładem może być tutaj sytuacja, kiedy w wybranym miejscu powierzchni chcemy stworzyć nowe, niewielkie detale, a aktualna konstrukcja siatki nam na to nie pozwala. Możemy wówczas wykorzystać do tego celu modyfikację Tessellate w połączeniu z modyfikacją Edit Mesh, nie jest to jednak jedyny i najszybszy sposób.




◄ ▼ **Przykład zastosowania modyfikacji Tessellate dla fragmentu obiektu. Jako narzędzie selekcji posłużyła tutaj modyfikacja Edit Mesh. Po lewej stronie widzimy kulę z zaznaczonymi pięcioma ściankami, a po prawej znajduje się kula z modyfikacją Tessellate korzystającą z wykonanej wcześniej selekcji.**

## Modyfikacja HSDS

Podczas modelowania, kiedy pracujemy tylko nad wybranymi fragmentami siatki, o wiele bardziej od Tessellate przydatna jest modyfikacja **HSDS**. Zawiera ona bowiem w swoim zestawie zarówno opcje pozwalające zwiększać rozdzielczość siatki, jak i opcje umożliwiające selekcję podobiektów i modelowanie powierzchni. Zaznaczyć trzeba jednak, że modyfikacja HSDS stworzona została przede wszystkim na potrzeby końcowego etapu modelowania, kiedy to na powierzchni dodajemy detale i wprowadzamy różnego rodzaju poprawki. Nie jest to typowe narzędzie do modelowania obiektów od podstaw.

Pod skrótem HSDS kryje się **Hierarchical SubDivision Surfaces**, czyli hierarchiczny podział powierzchni. W praktyce modyfikacja HSDS to tworzenie kolejnych poziomów podziału

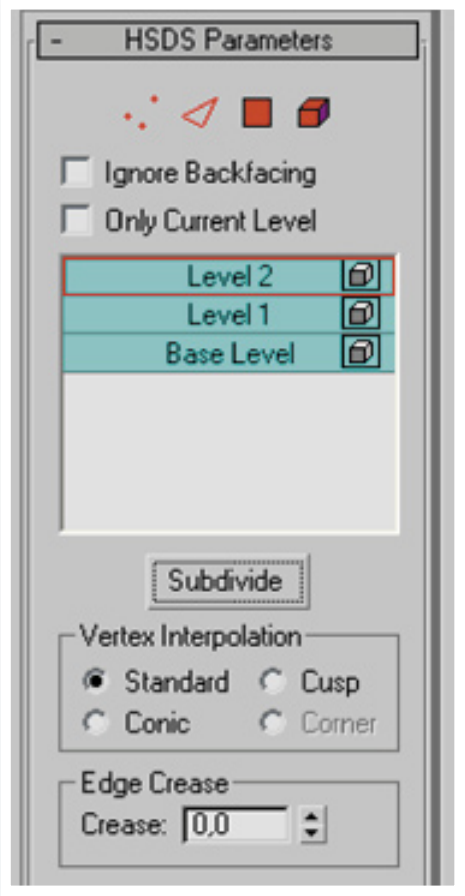
i modelowanie na poszczególnych poziomach złożoności siatki. Zobaczmy zatem jak wygląda praca z tą modyfikacją i jak funkcjonują najważniejsze opcje HSDS.

## Selekcja i transformacja

Na szczycie rolety **HSDS Parameters** znajdują się opcje selekcji podobiektów. Umieszczone są tutaj cztery przyciski **Vertex**, **Edge**, **Polygon** i **Element**, które pozwalają zdecydować na jakim poziomy struktury chcemy pracować, czy mają to być wierzchołki, czy też krawędzie lub ścianki. Po włączeniu jednej z tych opcji możemy zaznaczać kursorem podobiekty w oknie widoku i transformować zestaw selekcji za pomocą opcji Move, Scale i Rotate.

Opcja **Ignore Backfacing** pojawiała się już w naszych wcześniejszych opisach, pozwala ona wykluczyć z selekcji te wierzchołki i ścianki, które są odwrócone do nas tyłem.

Z kolei opcja **Only Current Level** jest przydatna w momencie kiedy już podzielimy naszą siatkę. Za jej pomocą możemy wówczas ukryć podobiekty, które nie należą do aktualnie zaznaczonego poziomu podziału. Jest to przydatne zwłaszcza podczas pracy z bardziej złożonymi modeli, gdyż poprawia widoczność w oknie oraz przyspiesza pracę programu.

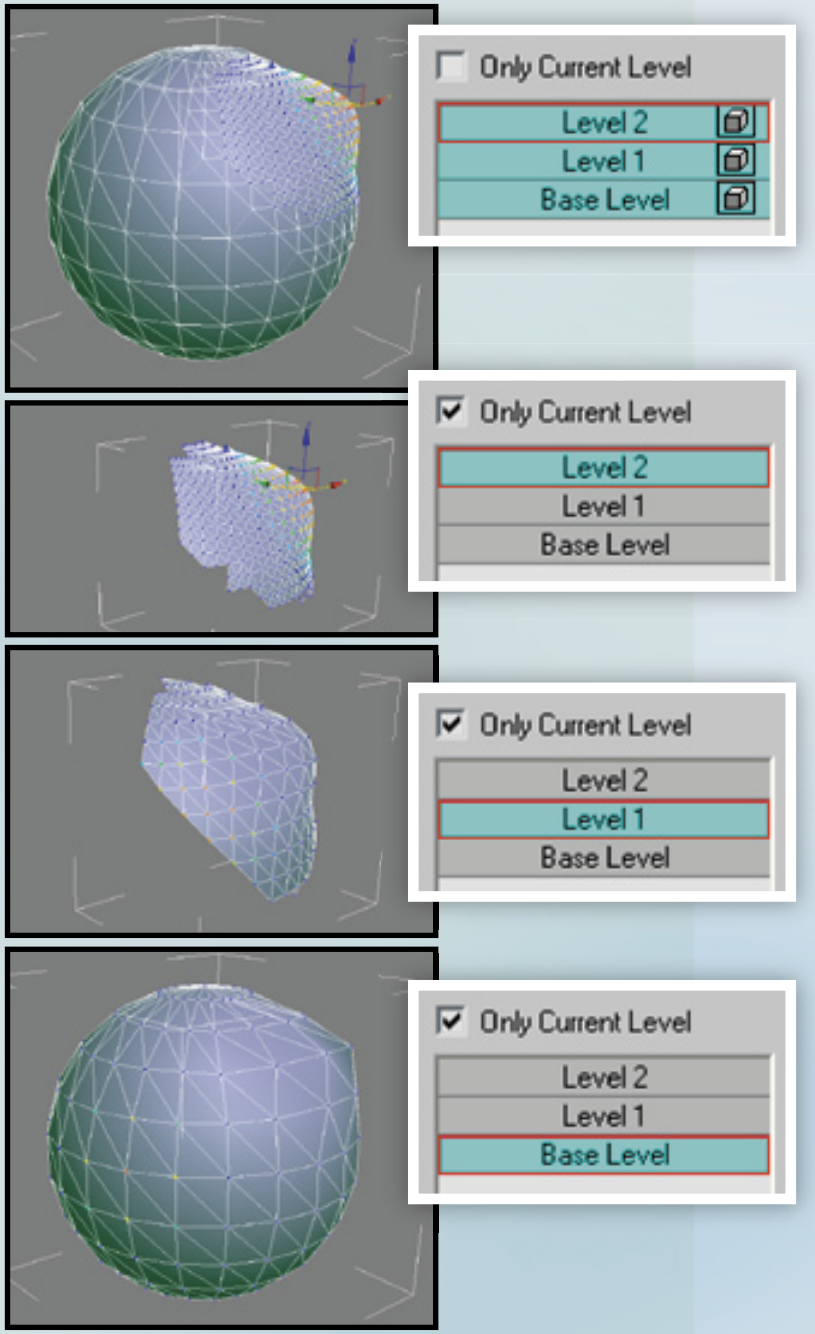


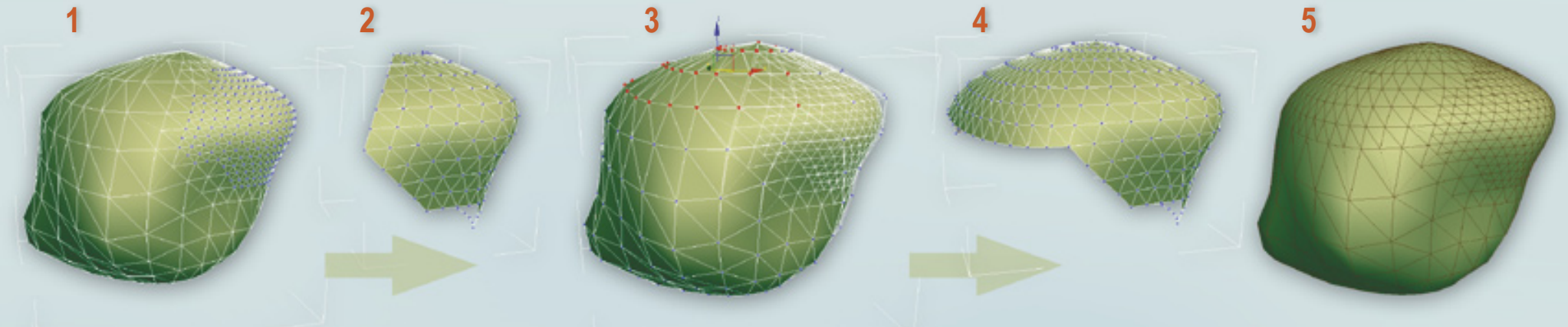

▲ Roletę HSDS Parameters znajdziemy na bocznej listwie, w zestawie opcji modyfikacji HSDS. Grupa Vertex Interpolation dostępna jest tylko na poziomie wierzchołków, a grupa Edge Crease dostępna jest tylko na poziomie krawędzi.

## Podział ścianek i poziomy

Przechodzimy teraz do opcji podziału siatki. W rolecie HSDS Parameters znajduje się lista poziomów podziału,

▼► Po prawej stronie ilustracje przedstawiające działanie opcji Only Current Level. Widzimy tutaj kulę, której fragment podzielony został dwukrotnie, a zatem na liście Subdivision Stack znajduje się poziom Base Level oraz dwa utworzone poziomy podziału Level 1 i Level 2. Jeżeli opcja Only Current Level jest wyłączona, to wszystkie trzy poziomy są wyświetlane w oknie widoku (o ile w okienku obok poziomów znajduje się ikona sześcianu). Po włączeniu opcji Only Current Level wyświetlane mogą być tylko ścianki poziomu aktualnie zaznaczonego w Subdivision Stack. Na górnej ilustracji wyświetlane są wszystkie poziomy, natomiast na trzech ilustracjach poniżej widoczny jest tylko wybrany level.

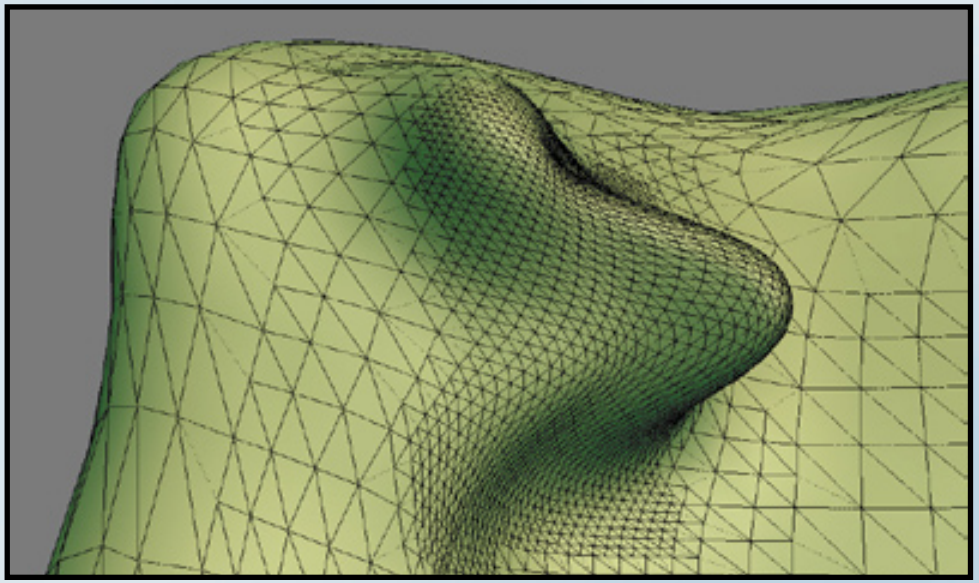





◄ ▼ **Powyższy ciąg ilustracji ukazuje sytuację, w której dochodzi do połączenia podziału. Pierwszy krok to dwukrotny podział fragmentu jakiegoś wykrzywionego obiektu - na liście Subdivision Stack znajdują się poziomy: Base Level, Level 1 i level 2. Drugi krok przedstawia nam siatkę poziomu Level 1 (za pomocą opcji Only Curent Level). W trzecim kroku, na poziomie Base Level dokonano selekcji wierzchołków, które wcześniej nie były modyfikowane za pomocą opcji Subdivide. Co się stanie, jeżeli teraz na poziomie Base Level klikniemy na przycisk Subdivide? Efekt takiej operacji ukazuje krok czwarty - jest to siatka poziomu Level 1. Widzimy, że do istniejącego wcześniej podziału (krok 2) dodany został podzielony w trzecim kroku fragment siatki. Krok piąty - końcowy efekt tych operacji.**

która określana jest jako Subdivision Stack. Zaraz po przypisaniu modyfikacji znajduje się na niej tylko jedna pozycja, jest to **Base Level**, czyli początkowa postać modelu. Po prawej stronie każdej nazwy poziomu znajduje się pole z ikonką przedstawiającą sześcian, które pozwala włączyć lub wyłączyć widoczność siatki danego poziomu podziału. Ikona ta jest przydatna dopiero kiedy na liście znajdują się co najmniej dwie pozycje. Dzięki tej opcji możemy wyświetlać bardziej złożoną siatkę na niższym poziomie. Jeżeli przykładowo mamy trzy poziomy i zaznaczymy poziom Base Level, to jeżeli w dwóch powyższych poziomach włączona jest opcja wyświetlania, to w oknie widoku będą widoczne wszystkie podziały siatki. Ikony z małymi sześcianami są niedostępne jeżeli mamy włączoną opcję Only Current Level.

Poziomy podziału ułożone są na liście w sposób hierarchiczny. Na samym dole znajduje się podstawowy poziom Base, a nad nim ułożone są kolejne poziomy podziału, aż do pierwszego na liście - najbardziej złożonego podziału siatki. Aby wykonać podział dokonujemy selekcji w oknie widoku, a następnie klikamy na klawisz Subdivide, który dzieli ścianki należące do aktualnego zestawu selekcji i dodaje nowy poziom na listę Subdivision Stack. Warto zwrócić uwagę na to, że jeżeli zaznaczymy jeden z dodanych poziomów podziału na liście, to w tym momencie mamy dostęp tylko do podobiektów należących do podzielonego obszaru.



▲ **Modyfikację HSDS wykorzystywać możemy w różnych sytuacjach. Narzędzie znakomicie sprawdza się podczas tworzenia wszelkiego rodzaju mniejszych detali w wybranych punktach siatki, zwłaszcza podczas pracy z modelami organicznymi lub wszelkiego rodzaju gładkimi kształtami. HSDS wykorzystać możemy również jako szybki sposób na pozyskiwanie siatek na potrzeby innych konstrukcji.**

## Siatka pomocnicza i ostre krawędzie

Za pomocą modyfikacji HSDS możemy dokonać podziału całej siatki obiektu - zaznaczając wszystkie podobiekty, jednak narzędzie to stosuje się głównie w przypadkach kiedy dzielimy tylko fragment siatki (do podziału całości wygodniej jest skorzystać z opisywanej wcześniej modyfikacji Tessellate).

Praca na poziomach struktury modyfikacji HSDS oparta jest na korzystaniu z siatki pomocniczej, która oddziałuje na siatkę obiektu. Poszczególne wierzchołki należące do obiektu przemieszczają się pod wpływem tej właśnie siatki pomocniczej, którą możemy dowolnie modelować w oknie widoku. Siatka pomocnicza jest wyraźnie widoczna zwłaszcza jeżeli obiekt posiada kilka poziomów podziału, a my pracujemy na jednym z najniższych leveli.

Pod listą Subdivision Stack znajduje się grupa opcji **Vertex Interpolation**, która dostępna jest tylko w czasie pracy na poziomie Vertex. Znajdują się w tej grupie cztery opcje wyboru: **Standard**, **Conic**, **Cusp** i **Corner**. Każda z nich przydatna jest w sytuacji, gdy przemieszczamy wierzchołek na poziomie niższym niż najwyższy poziom podziału. Jeżeli przykładowo posiadamy obiekt z kilkoma poziomami na liście Subdivision Stack i przemieszczamy wierzchołek korzystając z opcji Standard (na Base Level), to możemy zauważyć że wierzchołek siatki obiektu nie przylega do siatki pomocniczej. Jeżeli natomiast włączymy opcję Conic lub Cusp to siatka obiektu przylega do siatki pomocniczej wraz z przesuwaniem wierzchołka.

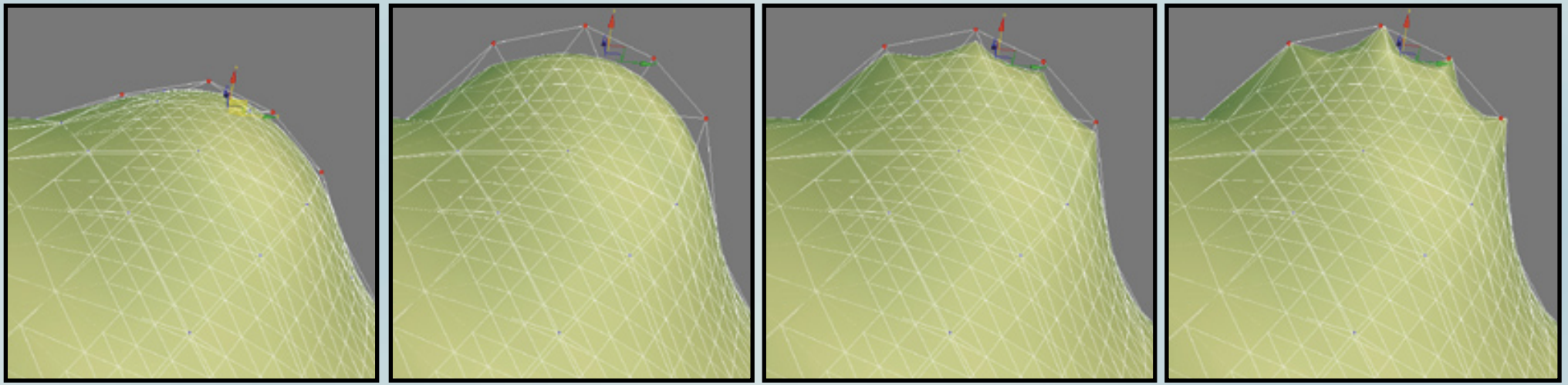
Opcja Corner dostępna jest tylko kiedy zaznaczamy skrajne wierzchołki otwartej krawędzi - przyciąga do siatki pomocniczej wierzchołki krawędzi, która została zaokrąglona w trakcie dokonywania podziału.

U dołu rolety HSDS Parameters znajduje się parametr **Crease**, który jest dostępny tylko na poziomie Edge. Wartość określa stopień przyciągania krawędzi obiektu do siatki pomocniczej. Im większa wartość, tym krawędź siatki modelu znajduje się bliżej aktualnie zaznaczonych podobiektów. Dzięki tej opcji możemy wyostrzać krawędzie obiektu w miejscach gdzie dokonujemy transformacji na poziomie Edge (na nie najwyższym poziomie podziału). Domyślnie przyjmuje on wartość 0, co oznacza że krawędź obiektu nie jest w żadnym stopniu przyciągana do krawędzi siatki pomocniczej.

## Opcje zaawansowane

Druga roleta - **Advanced Options** zawiera zestaw dodatkowych opcji, które w niektórych sytuacjach mogą okazać się bardzo przydatne. Opcja

**▼► Na ilustracjach poniżej widzimy jak zachowują się wierzchołki dla trzech opcji Vertex Interpolation. Pierwsza ilustracja po lewej ukazuje selekcję czterech wierzchołków na jednym z niższych poziomów podziału, jeżeli dobrze się przyjrzymy możemy zauważyć tutaj siatkę pomocniczą, która odbiega nieco od kształtu powierzchni. Trzy kolejne ilustracje ukazują przesunięcie tych wierzchołków z zastosowaniem trzech metod Vertex Interpolation (Standard, Conic i Cusp).**



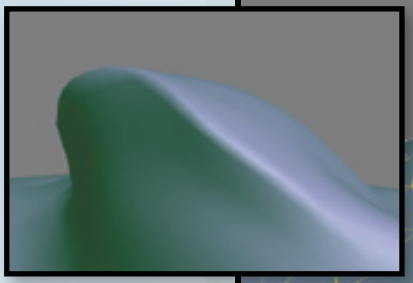
**Force Quads** pozwala zdecydować jaką siatkę wyjściową chcemy otrzymać, czy ma to być siatka zbudowana z trójkątów, czy też z czworokątów. Dostępna jest ona tylko dla obiektów, które posiadają jakąkolwiek ściankę nie zbudowaną z czterech krawędzi.


▲ **Roleta Advanced Options.**

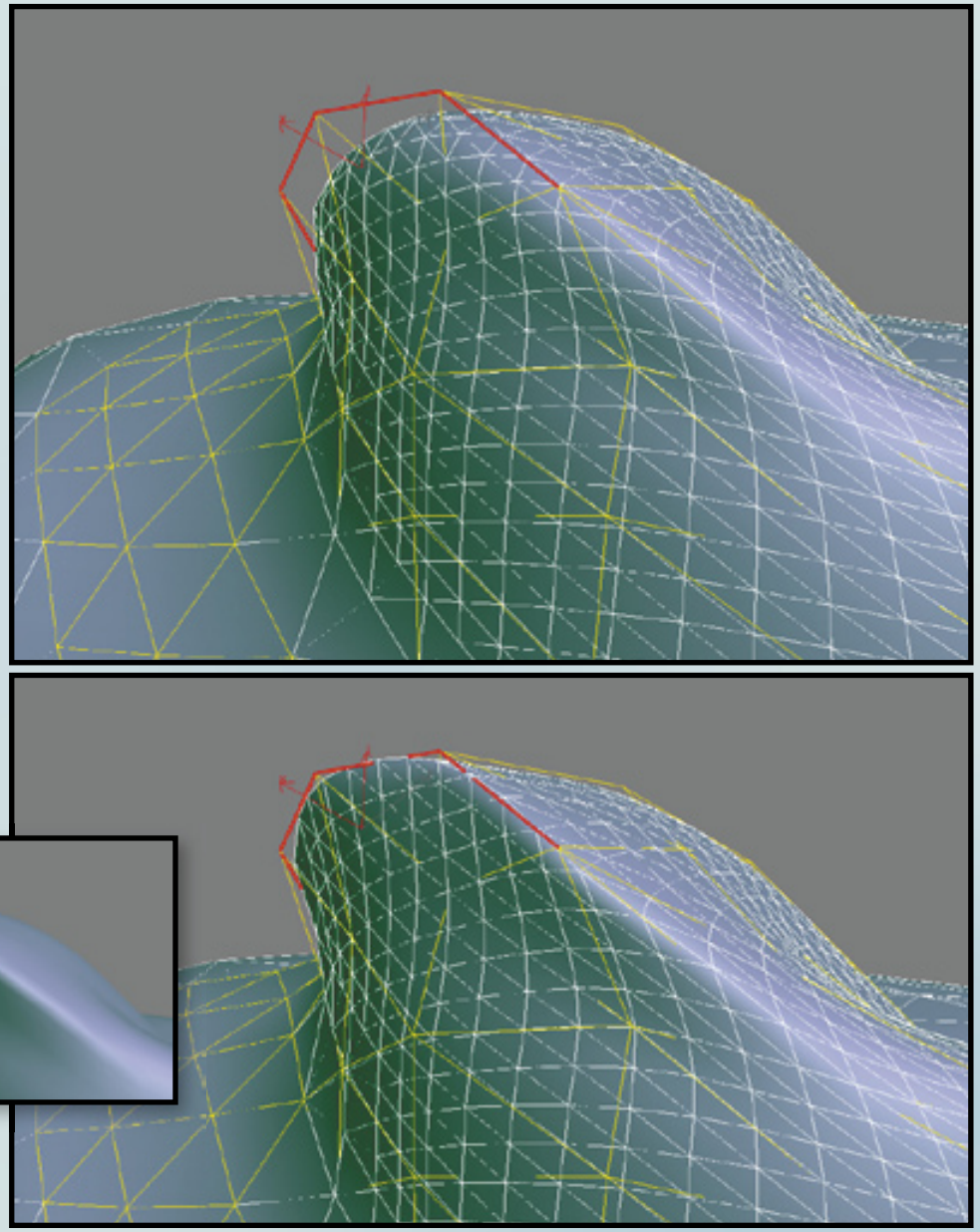
Jeżeli np. przypiszemy modyfikację do zwykłego box'a to opcja będzie wygaszona. Domyślnie opcja jest wyłączona, co oznacza że modyfikacja HSDS konwertuje wszystkie ścianki tak, że otrzymujemy siatkę zbudowaną z trójkątów. Natomiast opcja Force Quads zamienia wszystkie trójkątne ścianki na czworokątne polygony, poprzez wstawienie w odpowiednich miejscach nowych krawędzi. Modyfikacja Skin najlepiej funkcjonuje właśnie z czworokątami, dlatego też przeważnie wskazane jest zaznaczenie tej opcji zaraz po przypisaniu modyfikacji.

Opcja **Smooth Result** nadaje jedną grupę wygładzenia wszystkim ściankom obiektu, a parametr **Material ID** (dostępna jedynie na po zaznaczeniu co najmniej jednej scianki) przypisuje numer ID materiału. Przycisk **Hide** pozwala ukrywać zaznaczone polygony, a przycisk **Unhide All** przywraca wszystkie ukryte ścianki. Natomiast za pomocą klawisza **Delete Polygon** usuwamy zaznaczone ścianki. Opcje ukrywania i usuwania są dostępne tylko podczas pracy na poziomach struktury Polygon i Element.

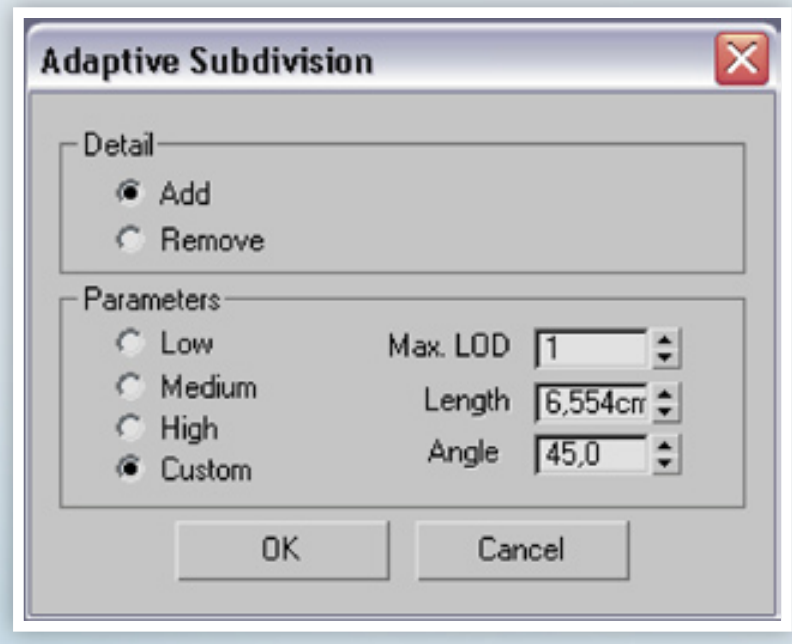
Na samym dole znajdziemy przycisk **Adaptive Subdivision**, który otwiera niewielkie menu z opcjami pozwalającymi dodatkowo podzielić siatkę obiektu lub też ją uprościć. W grupie Detail wybieramy pomiędzy opcją **Add** i **Remove**. Pierwsza z tych opcji oznacza dodatkowy podział siatki, natomiast druga opcja upraszcza powierzchnię.



W grupie Parameters ustalamy natomiast zakresie działania opcji



▲ **Zastosowanie parametru Crease. Na górnej ilustracji zaznaczonych jest kilka krawędzi jednego z niższych poziomów Subdivision. Wratość parametru Crease wynosi tutaj 0. U dołu widzimy natomiast jak zmienia się kształt siatki obiektu kiedy wartość Crease zwiększymy do maksimum (1.0).**




◄ **Klawisz Adaptive Subdivision otwiera dodatkowe menu, w którym możemy dzielić lub upraszczać siatkę obiektu.**

Adaptive Subdivision. Znajdują się tutaj trzy domyślne ustawienia (Low, Medium, High) oraz opcja **Custom**, która umożliwia samodzielne określenie wartości parametrów.

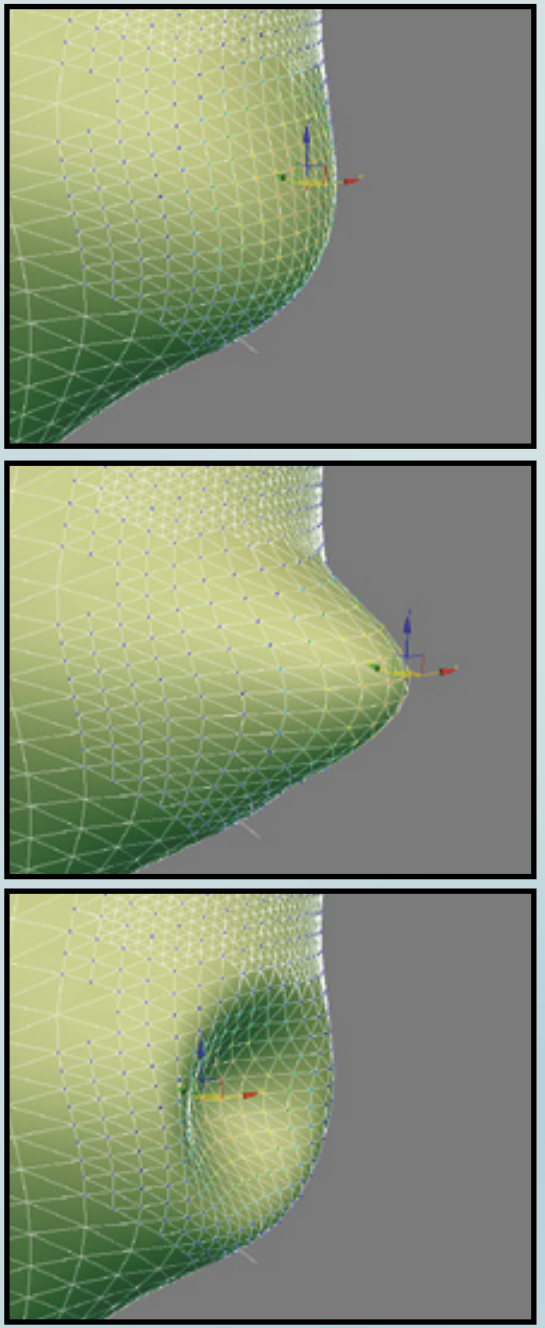
Parametr **Max LOD** definiuje największą liczbę poziomów podziału jakie mogą zostać dodane w wyniku przeprowadzenia operacji Add (dla opcji Remove jest on niedostępny).

Parametry **Length** i **Angle** są to wartości, które musza być spełnione aby doszło do podziału siatki. Wartość Length to maksymalną długość krawędzi, a wartość Angle to maksymalny kąt pomiędzy krawędziami wychodzącymi z jednego wierzchołka. Im mniejsza wartość Length i Angle, tym większy staje się zakres podziału. Z Adaptive Subdivision najczęściej korzysta się w celu wygładzenia lub uproszczenia siatki, pod koniec pracy z modyfikacją HSDS, kiedy to już nie zamierzamy wykonywać żadnych czynności za pomocą innych opcji.

## Miękka selekcja

Modyfikacja HSDS posiada również roletę **Soft Selection**, która pozwala na dokonywanie miękkiej selekcji w oparciu o zasięg i krzywą określającą siłę oddziaływania. Jest to niezwykle przydatny zestaw opcji, który umożliwia modelowanie powierzchni i wprowadzanie różnego rodzaju przekształceń. Z opcjami z tej rolety zapoznaliśmy się już we wcześniejszych numerach magazynu, dlatego tym razem je pominiemy.

I to już wszystko na temat HSDS. Modyfikacja ta nie oferuje wielkiej ilości opcji, ale mimo to może być bardzo przydatna w trakcie modelowania obiektów. Praca z listą podziałów umożliwia detalizowanie powierzchni, a korzystać możemy przy tym z różnych poziomów złożoności dzięki liście Subdivision Stack. Połączenie podstawowych narzędzi selekcji z opcjami podziału ścianek znakomicie sprawdza się w zagęszczaniu fragmentów siatki. ■



▲► **Dużym plusem modyfikacji HSDS jest to, że w skład jej opcji wchodzi roleta Soft Selection. Nie musimy zatem przypisywać osobnej modyfikacji jeżeli chcemy skorzystać z miękkiej selekcji. Opcje z tej rolety umożliwiają modelowanie powierzchni obiektu i są bardzo przydatne zwłaszcza kiedy mamy do czynienia z modelami organicznymi. Na ilustracjach po lewej widzimy jak poprzez przesunięcie tylko jednego wierzchołka zmienia się kształt całego fragmentu siatki.**

# Torturr


**ARTUR SADŁOS** Concept Artist & Illustrator

www.artoftorturr.blogspot.com
torturr@gmail.com

▶ Alice Wars

■ Niewyobrażone Sny Nienarodzonego Płodu
Praca na bitwę „Małe jest piękne" organizowaną przez max3d.pl.

**WARP: Jak to się stało, że zająłem się grafiką:**

**ARTUR SADŁOS:** Dawno temu, około siedmiu lat wstecz obejrzałem po raz pierwszy Dragon Ball, anime na podstawie mangi Akiry Toriyamy. Wessało mnie zupełnie. Tak w międzyczasie bazgrałem sobie postacie i jakieś małe historyjki ze świata Dragon Ball. Tak bardzo mi się to spodobało, że postanowiłem że kiedyś będę zajmował się tym na poważnie. I poleciało. Dużo rysowałem, ćwiczyłem i po ukończeniu liceum wybrałem Instytut Sztuk Pięknych w Kielcach (nie czułem się na siłach na ASP). Na drugim roku zaopatrzyłem się w komputer i odkryłem Photoshopa, cudowny program. Kolorowałem w nim komiksy i robiłem badziewne fotomontaże :) Prawdziwym przełomem był zakup tabletu. Dopiero to narzędzie umożliwiło mi pełne wykorzystanie dobrodziejstw digital artu. I teraz się pilnie uczę i ćwiczę, żeby kiedyś zbliżyć się do poziomu mistrzów ;)

■ Battle Babe
Jedna z prac dla Liberty of Mind.

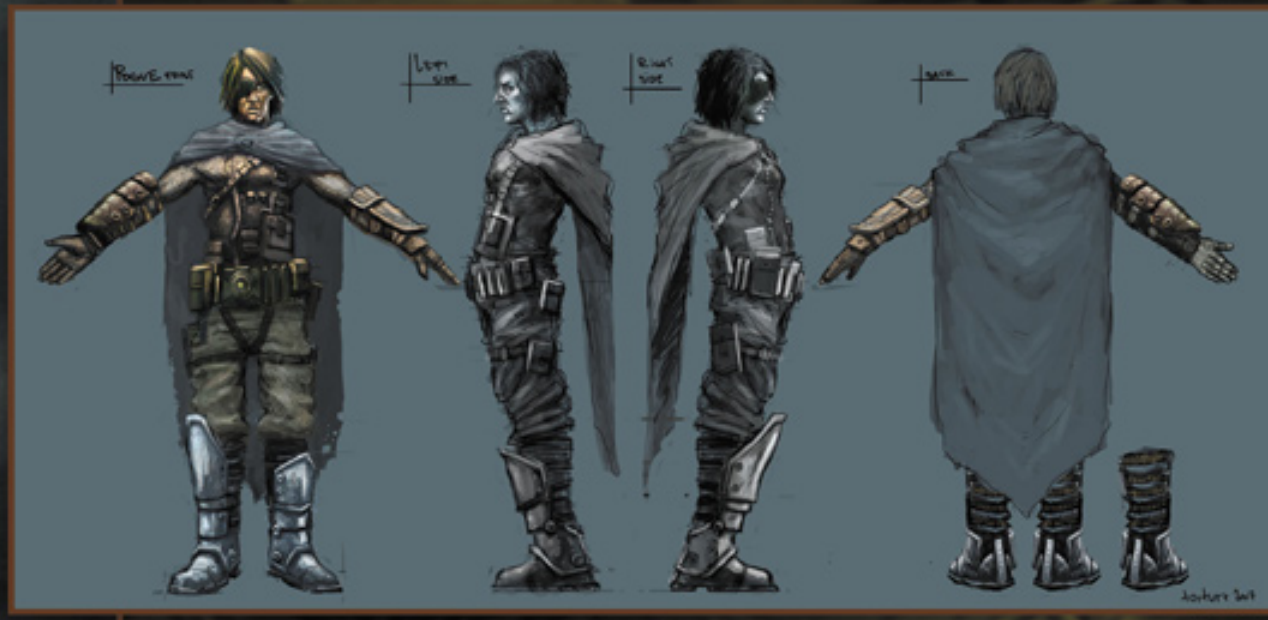
## w: Moje największe marzenie:

**AS:** Apartament w Tokio, piękna żona i święty spokój ;) A tak w temacie to praca w przemyśle filmowym lub gier video. Jako concept designer, ilustrator czy storyboard artist. Marzy mi się duża produkcja sci-fi lub fantasy i praca wśród utalentowanych ludzi którzy podzielają tę samą pasję.

## w: Plany na najbliższą przyszłość:

**AS:** Osiągnięcie choć części tego co opisałem powyżej :)

▼ Rogue concept



▼ Gringo Turtle

▼ Flying Ukap

▼ Karate Gorilla

▼ Vampire Warrior Bitch

### W: Pierwsza praca 2D/3D:

**AS:** Roboty rysowane w książeczce zdrowia mojego dziadka. Słyszałem, że w przychodni mieli ubaw. Miałem kilka latek wtedy, nie że jakoś niedawno...

### W: Ulubiony film/gra komputerowa:

**MO:** Dużo tego. To są pozycje obowiązkowe: trylogia Indiana Jones (wkrótce tetralogia, Steven nie spieprz tego!), trylogia Back To The Future, trylogia Star Wars. Cenię Spielberga, Ridleya Scotta, Camerona jak i Bergmana, Felliniego czy Kieślowskiego. Od klasyki po współczesne kino rozrywkowe i tzw. ambitne. Jakoś nie czuję konfliktu w sobie oglądając amerykańską komercję i niszowe kino autorskie, lubię kino zwyczajnie.

Jeśli chodzi o gry to zdecydowanie seria Final Fantasy. Szczególne miejsce ma część siódma za ogrom radochy jaki mi dała. Za to dziewiątka powaliła mnie i do dziś powala genialnym designem świata gry. Chciałbym kiedyś móc pozwiedzać taki zamek Lindblum, mój ideał. Fajna jest też gierka Tombi! którą do dziś sobie odpalam na konsoli. Jeśli chodzi o nowości to nie jestem na bieżąco. Za dużo pracy i za mało czasu na gry. W wolnych chwilach wolę oderwać się od kompa i spędzić czas z dziewczyną poza domem.

▲ Dog Fight
**Okładka książki „Bo to jest wojna, rzeź i rąbanka“ wykonana dla wydawnictwa Fabryka Słów.**

### w: Najtrudniejsze dla mnie jest:

**AS:** Kończenie prac. Zawsze gdy jestem już tak gdzieś na etapie 60% skończonego obrazka, nachodzi mnie ochota na rozpoczęcie nowego. Uwielbiam proces projektowania i budowania koncepcji, szybkie szkice i nakreślanie klimatu. Zawsze wolałem szkice od skończonych prac.

### w: Idealny projekt, w którym chciałbym uczestniczyć:

**AS:** Jakiś projekt Lucasa i Spielberga. Cokolwiek od tych panów hy hy.

### w: Mój autorytet/idol/ ulubiony artysta:

**AS:** Leonardo da Vinci, Rembrandt, Craig Mullins, Justin Sweet, Tim Burton, H.R. Giger, Śledziu, John Williams, Banksy, Akira Toriyama, Yoshitaka Amano i tysiące innych wspaniałych ludzi którzy dają mi motywację do pracy.

► Esteban Padre De Nascimento
Jedna z postaci z gry Battle Rage.

## w: Moja aktualna maszyna:

**AS:** Stary słaby pecet, tablet Wacom Intuos A5 Wide i 21 calowy LCD panorama. Reszta się nie liczy.

## w: Wkurza mnie:

**AS:** Przeziębienia jesienią, wykańczają mnie ostatnio. No i moje wrodzone lenistwo i brak organizacji.

## w: Aktualnie pracuję nad:

**AS:** Dla Liberty Of Mind nad niezliczoną ilością postaci kreskówkowych, do banerów i nad logosami dla witryn. Dla Teyonu nad grą Battle Rage oraz nad platformerem 2D Quasimodo dla Play Publishing. Czasami robię okładki dla Fabryki Słów. Plus różne drobne zlecenia, zależy co i dla kogo się trafi. ■

■ Witch By Proffesion

# Battle Rage

## PROLOGUE 1 & 2

▲◄ Prologue 1 i Prologue 2
Dwie ilustracje promocyjne wykonane na potrzeby gry Battle Rage, która powstaje w Destan Entertainment (www.battle–rage.com).

# WARP

BLACK CAT TEAM