nost je v jeho syntetickém původu, jedná se o jazyk umělý. Programovací jazyk je přednostně zkonstruován pro definici známého a pochopeného. V případě neznámých nebo nepoznaných veličin je programovací jazyk víceméně k ničemu.

Chceme-li komunikovat se strojem, musíme tedy svůj způsob vyjadřování přizpůsobit logicky dokonalému jazyku – vnitřní logice fungování stroje. Počítač není navržen k tomu, aby něčemu rozuměl. Počítač je navržen k řešení jasně definovaných otázek. Tato příručka se pokusí srozumitelnou formou popsat jeden z možných způsobů, jak si takový jazyk osvojit a potažmo způsob uvažování, který vede k jasné definici problému. Hovoříme-li o programování, máme na mysli proces tvorby jisté logické struktury. Osvojení si programování spočívá ve schopnosti definovat problém nebo jasně formulovat otázku tak, aby na ni stroj mohl odpovědět.

Vtip spočívá v tom, že ovládneme-li formálně jazyk určený stroji, můžeme prostřednictvím tohoto stroje hovořit i ke člověku, tj. popisovat i pocity z rozkvetlých luk.

4.4 Volba vhodného jazyka

Ve výpočetní technice se nachází celá škála programovacích jazyků i prostředí. Tyto jazyky mají svoji genezi a byly historicky vyvíjeny především počítačovými odborníky. Jejich dokonalost lze těžko ocenit z vnějšího pohledu, a to právě z důvodu jejich konstrukce, která odpovídá a částečně podléhá určitým účelům, ke kterým byly tyto jazyky původně navrženy. Celistvý pohled na vývoj programovacích jazyků zde není možné obsáhnout. Programovací jazyk dle historického vývoje můžeme rozdělit na dvě skupiny, jazyky imperativní a objektově orientované. Toto základní rozdělení programovacích jazyků popisuje dva rozdílné přístupy v popisu událostí.

Processing se svou stavbou na základech *Javy* řadí k objektově orientovaným jazykům. Programovací jazyk *Java* je relativně mladým jazykem. Historie tohoto jazyka sahá přibližně do roku 1995. Mezi jeho hlavní výhody, ze kterých i *Processing* velmi těží, jsou zejména důraz na multiplatformnost a relativní jednoduchost syntaxe. Příbuznými jazyka *Java*

jsou další objektově orientované programovací jazyky jako *C++*, *Perl*, *Object Pascal*, *Visual Basic*, *C#*, *PHP*, *Python*, *Ruby*, *Smalltalk*, nebo *Lisp*.

Rozdíl mezi imperativním pojetím programování a objektově orientovaným pojetím spočívá především v logice kódu a jeho následném uspořádání. Imperativní jazyk se dá považovat za předchůdce objektově orientovaného pojetí. Nedá se obecně říci, který přístup je výhodnější, jedná se o dvě rozdílná paradigmata. Dnes mezi nejpoužívanějšími programovacími jazyky masivně převládá objektově orientované paradigma.

Rozdílné pohledy lze ilustrovat na popisu nějakého jevu. Jev se dá popsat různými způsoby. Jedna perspektiva bude více hovořit o aktérech jevu, ty rozdělí do objektů, jejich vlastností a možností. Druhá perspektiva popíše celistvou situaci jako sérii událostí, které je možné v tomto pořadí kdykoli zopakovat.

V posledních přibližně dvaceti letech se mezi programovacími jazyky postupně objevuje tendence po větší srozumitelnosti a potažmo zjednodušení programování jako takového. Programování v této koncepci již není jazykem odborníků, ale je demokraticky přístupné širší veřejnosti z rozmanitých – prioritně netechnických oborů.

Tato tendence postupně dala vzniknout celé rodině programovacích jazyků, které se snaží přiblížit potenciál výpočetní techniky netechnickým oborům, v neposlední řadě i oborům výtvarným. V technických kruzích je ovšem již sama disciplína psaní programů považována za tvůrčí činnost. V pojetí výtvarného umění dnes převažuje pohled na programování jako na velmi technickou zdatnost, rigidní a notně limitovanou.

Jazyk, který je nutně limitován svou nezbytnou formální dokonalostí, ovšem nemusí nutně limitovat svého uživatele ve sdělení. Uživatel se musí k uskutečnění takové zdárné komunikace přizpůsobit stroji.

4.5 Proč zrovna Processing?

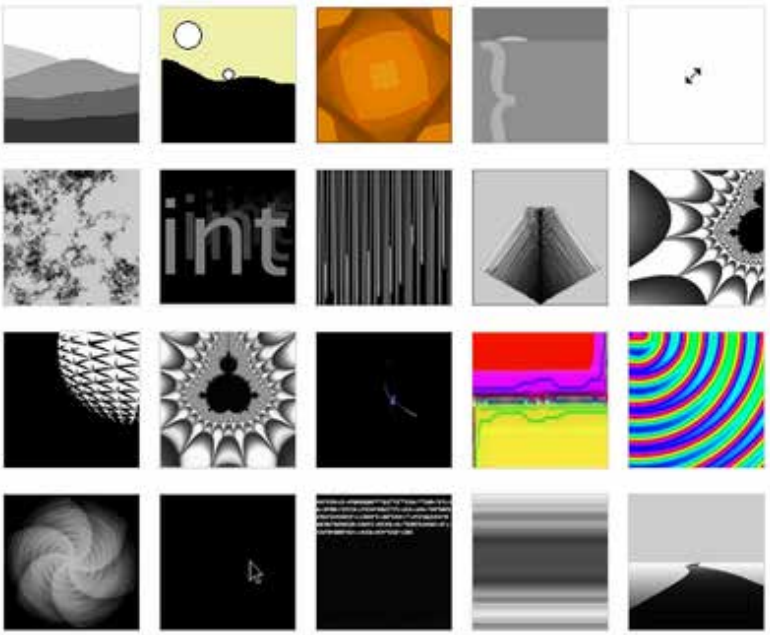
Processing je vhodný nástroj pro první experimenty s programováním z několika důvodů.

Prvním důvodem je jednoduchost pro uživatele, *Processing* se snaží začátečníky co nejméně zatížit zbytečnými informacemi. *Processing* prezentuje programování jako přístupnou a zvladatelnou schopnost. Dosahuje

toho především rychlou odezvou vizuálního prostředí. Jednoduše řečeno, v momentu, kdy spustíme kód, vidíme okamžitě jeho výsledek. Taková ilustrativnost je v začátcích podle mého názoru téměř nezbytná.

V *Processingu* lze velmi krátkým kódem programovat sofistikované programy. Pro příklad zde uvedu soutěž nazvanou *Tiny Sketch* pořádanou portály *rhizome.org* a *openprocessing.org*. *Processing* dokáže být natolik úsporný, že pomocí maximálně dvou set znaků byli tvůrci schopni naprogramovat svébytné programy.

Máte-li počítač u sebe a jste-li připojeni k internetu, vřele doporučuji si nyní projít aktuální podobu stránky openprocessing.org.



Snad největší předností pro začínající programátory je výtečná dokumentace a velmi přívětivá komunita lidí poskytující neúnavně rady začátečníkům.

4.6 Tvorba softwaru

Programy jsme v současnosti zvyklí spíše využívat než sami vytvářet. Tvoření programu je náročný proces a tvorba uživatelsky přátelského prostředí je velmi složitá.

Processing se svým způsobem neliší od žádného jiného programu, který běžně využíváme. Jde o sadu příkazů a programovací prostředí, které nám dovoluje určitou formou vytvářet svébytný program. I když se již jedná o programování, nelze jej běžně zaměňovat s klasickou tvorbou vyspělého nástroje.

Zde si musíme uvědomit, že náš potenciální výsledek – program bude vždy spíše banální v porovnání například se samotným prostředím *Processingu*. Processingový kód je výčet všech možností, které můžeme při tvorbě našeho programu využít. Obecně se dá říci, že *Processing* je nástroj pro tvorbu speciálních nástrojů. Výsledek našeho programování bude vždy pravděpodobně obsahovat poměrně větší část *Processingu* samotného.

Je dobré od začátku pochopit, k čemu lze *Processing* využít a k čemu jej opravdu využívat chceme. *Processing* se především hodí k rychlé tvorbě programu, ověření teze nebo spontánního nápadu. Pro podobné programování se také vžil pojem „rapid engineering“. Tvorba dospělejších nástrojů není sice nemožná, obecně ale platí, že *Processing* bude svými zkrácenými zápisy a jednoduchostí prostředí velmi nápomocný v začátcích.

Processing se svou konstrukcí zejména hodí pro práci s obrazem, a tedy obrazu bude také věnována nejrozsáhlejší část této knihy. Výstupy z *Processingu* lze nadále zpracovávat, *Processing* tak může sloužit například jako mezičlánek ve výrobním procesu.

Dnes je *Processing* využíván hlavně grafickými designéry, designéry užitého umění, tvůrci webových aplikací, softwarovými návrháři, vizuálními umělci a umělci, kteří se věnují instalacím v prostoru nebo živé projekci. K *Processingu* si ovšem nacházejí cestu i více technické obory, zejména architekti, badatelé v oboru přírodních věd, statistici a všeobecně tvůrci softwaru. Software nachází využití i v oborech zabývajících se humanitními vědami, jako například v sociologii nebo jazykovědě.

Processing se neomezuje na jeden obor, svou koncepcí spíše nabízí společnou platformu pro mezioborovou komunikaci.